UNIVERSIDAD NACIONAL DE PIURA

Facultad de Ciencias

Escuela Profesional de Ingeniería Electrónica y Telecomunicaciones

TESIS

"DISEÑO E IMPLEMENTACIÓN DE UN PROTOTIPO DE ALIMENTADOR AUTOMÁTICO PARA ANIMALES DOMÉSTICOS UTILIZANDO UNA PLATAFORMA EN MODULO ELECTRONICO"

Presentado por:

Br. Cajo Sotero Jesús Alberto

Br. Cardoza Atoche Víctor Raúl

PARA OPTAR EL TÍTULO PROFESIONAL DE:

INGENIERO ELECTRÓNICO Y TELECOMUNICACIONES

Línea de investigación: Instrumentación y Control

Piura - Perú

2019

UNIVERSIDAD NACIONAL DE PIURA

Facultad de Ciencias

Escuela Profesional de Ingeniería Electrónica y Telecomunicaciones

"DISEÑO E IMPLEMENTACIÓN DE UN PROTOTIPO DE ALIMENTADOR AUTOMÁTICO PARA ANIMALES DOMÉSTICOS UTILIZANDO UNA PLATAFORMA EN MODULO ELECTRONICO"

TESIS

PARA OPTAR EL TÍTULO PROFESIONAL DE:

INGENIERO ELECTRÓNICO Y TELECOMUNICACIONES

Ing. Edwin Arnaldo Ocas Infante
ASESOR

Ing. Daniel Alonso Flores Córdova
CO-ASESOR

Br. Cajo Sotero Jesús Alberto
AUTOR

Br. Cardoza Atoche Víctor Raúl
AUTOR

Piura – Perú

2019

DECLARACIÓN JURADA DE AUTENTICIDAD DE LA TESIS

Yo: Cajo Sotero Jesús Alberto con DNI N° 71322213, Bachiller de Escuela Profesional de Ingeniería Electrónica y Telecomunicaciones, de la Facultad de Ciencias y domiciliado en el AH cesar Vallejo MZ "C" lote "27" distrito 26 de octubre.

Celular: 947 685 107

Email: soteca92@gmail.com

DECLARO BAJO JURAMENTO: Que la tesis que presento es auténtica e inédita, no siendo copia parcial ni total de una tesis desarrollada, y/o realizada en el Perú o en el Extranjero, en caso contrario de resultar falsa la información que proporciono, me sujeto a los alcances de lo establecido en el Art. Nº 411, del código Penal concordante con el Art. 32° de la Ley Nº 27444, y Ley del Procedimiento Administrativo General y las Normas Legales de Protección a los Derechos de Autor.

En fe de lo cual firmo la presente.

Piura, noviembre del 2019

Cajo Sotero Jesús Alberto

Artículo 411.- El que, en un procedimiento administrativo, hace una falsa declaración en relación a hechos o circunstancias que le corresponde probar, violando la presunción de veracidad establecida por ley, será reprimido con pena privativa de libertad no menor de uno ni mayor de cuatro años.
Art. 4. Inciso 4.12 del Reglamento del Registro Nacional de Trabajos de Investigación para optar grados académicos y títulos profesionales –RENATI Resolución de Consejo Directivo Nº 033-2016-SUNEDU/CD

DECLARACIÓN JURADA DE AUTENTICIDAD DE LA TESIS

Yo: Cardoza Atoche Victor Raul con DNI N° 45198307 Bachiller de Escuela Profesional de Ingeniería Electrónica y Telecomunicaciones, de la Facultad de Ciencias y domiciliado en el Calle augusto B leguia N 410 el obrero Sullana

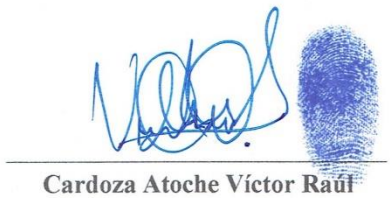
Celular: 985 942 917

Email: Cardoz_16@hotmail.com

DECLARO BAJO JURAMENTO: Que la tesis que presento es auténtica e inédita, no siendo copia parcial ni total de una tesis desarrollada, y/o realizada en el Perú o en el Extranjero, en caso contrario de resultar falsa la información que proporciono, me sujeto a los alcances de lo establecido en el Art. N° 411, del código Penal concordante con el Art. 32° de la Ley N° 27444, y Ley del Procedimiento Administrativo General y las Normas Legales de Protección a los Derechos de Autor.

En fe de lo cual firmo la presente.

Piura, noviembre del 2019



Cardoza Atoche Víctor Raúl

Artículo 411.- El que, en un procedimiento administrativo, hace una falsa declaración en relación a hechos o circunstancias que le corresponde probar, violando la presunción de veracidad establecida por ley, será reprimido con pena privativa de libertad no menor de uno ni mayor de cuatro años.
Art. 4. Inciso 4.12 del Reglamento del Registro Nacional de Trabajos de Investigación para optar grados académicos y títulos profesionales –RENATI Resolución de Consejo Directivo N° 033-2016-SUNEDU/CD

iv

UNIVERSIDAD NACIONAL DE PIURA

Facultad de Ciencias

Escuela Profesional de Ingeniería Electrónica y Telecomunicaciones

"DISEÑO E IMPLEMENTACIÓN DE UN PROTOTIPO DE ALIMENTADOR AUTOMÁTICO PARA ANIMALES DOMÉSTICOS UTILIZANDO UNA PLATAFORMA EN MODULO ELECTRONICO"

TESIS

PARA OPTAR EL TÍTULO PROFESIONAL DE:

INGENIERO ELECTRÓNICO Y TELECOMUNICACIONES

Dr. ANTENOR SEGUNDO ALIAGA ZEGARRA

PRESIDENTE DE JURADO DE TESIS

Dr. CARLOS ENRIQUE ARELLANO RAMIREZ

SECRETARIO DE JURADO DE TESIS

Ing. FRANKLIN BARRA ZAPATA MSc.

VOCAL DE JURADO DE TESIS

UNIVERSIDAD NACIONAL DE PIURA
FACULTAD DE CIENCIAS

"AÑO DE LA LUCHA CONTRA LA CORRUPCIÓN Y LA IMPUNIDAD"

ACTA DE SUSTENTACIÓN 074-2019-UI-FC-UNP

Los Miembros del Jurado Calificador que suscriben, reunidos para evaluar la Tesis denominada **"DISEÑO E IMPLEMENTACIÓN DE UN PROTOTIPO DE ALIMENTADOR AUTOMÁTICO PARA ANIMALES DOMÉSTICOS UTILIZANDO UNA PLATAFORMA EN MÓDULO ELECTRÓNICO"**, presentado por el Señor Bachiller **JESÚS ALBERTO CAJO SOTERO**, con el asesoramiento del **Ing. Edwin Arnaldo Ocas Infante** y co-asesor **Ing. Daniel Alonso Flores Córdova**; oídas las observaciones y respuestas a las preguntas formuladas, y de conformidad al Reglamento de Tesis para obtener el Título Profesional en la Facultad de Ciencias, lo declaran:

APROBADO (X) **DESAPROBADO** ( )

Con la mención de:

BUENO

(X) En consecuencia, queda en condición de ser ratificado por el Consejo de Facultad de Ciencias de la Universidad Nacional de Piura, y recibir el **TITULO PROFESIONAL DE INGENIERO ELECTRÓNICO Y TELECOMUNICACIONES.**

(X) En consecuencia, queda en condición de ser ratificado por el Consejo Universitario de la Universidad Nacional de Piura, y recibir el **TITULO PROFESIONAL DE INGENIERO ELECTRÓNICO Y TELECOMUNICACIONES**, después que el sustentante incorpore la sugerencia del Jurado Calificador.

Piura, 23 de noviembre del 2019.

Dr. ANTENOR SEGUNDO ALIAGA ZEGARRA
PRESIDENTE DE JURADO DE TESIS

Dr. CARLOS ENRIQUE ARELLANO RAMÍREZ
SECRETARIO DE JURADO DE TESIS

Ing. FRANKLIN BARRA ZAPATA, MS.c.
VOCAL DE JURADO DE TESIS

DIRECTOR

Campus Universitario - Urb.Miraflores S/N. Castilla
PIURA – PERU

Ampliar (Ctrl+0)

UNIVERSIDAD NACIONAL DE PIURA
FACULTAD DE CIENCIAS

"AÑO DE LA LUCHA CONTRA LA CORRUPCIÓN Y LA IMPUNIDAD"

ACTA DE SUSTENTACIÓN 075-2019-UI-FC-UNP

FACULTAD DE CIENCIAS

Los Miembros del Jurado Calificador que suscriben, reunidos para evaluar la Tesis denominada **"DISEÑO E IMPLEMENTACIÓN DE UN PROTOTIPO DE ALIMENTADOR AUTOMÁTICO PARA ANIMALES DOMÉSTICOS UTILIZANDO UNA PLATAFORMA EN MÓDULO ELECTRÓNICO"**, presentado por el Señor Bachiller **VÍCTOR RAÚL CARDOZA ATOCHE**, con el asesoramiento del **Ing. Edwin Arnaldo Ocas Infante** y co-asesor **Ing. Daniel Alonso Flores Córdova**; oídas las observaciones y respuestas a las preguntas formuladas, y de conformidad al Reglamento de Tesis para obtener el Título Profesional en la Facultad de Ciencias, lo declaran:

APROBADO (X) **DESAPROBADO** ( )

Con la mención de: Bueno

(X) En consecuencia, queda en condición de ser ratificado por el Consejo de Facultad de Ciencias de la Universidad Nacional de Piura, y recibir el **TITULO PROFESIONAL DE INGENIERO ELECTRÓNICO Y TELECOMUNICACIONES.**

(X) En consecuencia, queda en condición de ser ratificado por el Consejo Universitario de la Universidad Nacional de Piura, y recibir el **TITULO PROFESIONAL DE INGENIERO ELECTRÓNICO Y TELECOMUNICACIONES**, después que el sustentante incorpore la sugerencia del Jurado Calificador.

Piura, 23 de noviembre del 2019.

UNP

Dr. ANTENOR SEGUNDO ALIAGA ZEGARRA
PRESIDENTE DE JURADO DE TESIS

Dr. CARLOS ENRIQUE ARELLANO RAMÍREZ
SECRETARIO DE JURADO DE TESIS

Ing. FRANKLIN BARRA ZAPATA, MS.c.
VOCAL DE JURADO DE TESIS

DIRECTOR

Campus Universitario - Urb.Miraflores S/N. Castilla
PIURA – PERU

DEDICATORIA

Br. Cajo Sotero Jesús Alberto

Todo el trabajo que se ha desarrollado en esta tesis está dedicada en primer lugar a Dios, pues fue el quien me brindo salud y sabiduría para cumplir los objetivos que me propuse.

En segundo lugar está dedicada con mucho carriño a mis padres Elena y Ramiro que están a mi lado brindándome su mano amiga dándome a cada instante una palabra de aliento para llegar a culminar mi profesión. Me enseñaron lo que es la base de mi vida ser disciplinado y sin ella no hubiese alcanzando el éxito. También la dedico a mi querido abuelo Luis Flavio me enseño el control de mi temperamento, sé que no nos acompaña físicamente pero que desde el cielo me protege.

En tercero lugar la dedico a mis hermanas Noemí y Rosa por sus palabras y compañía, además a mi sobrino Thiago que con su sonrisa me anima a terminar este ensayo.

Y por último se la dedico a la Escuela profesional de Ingeniería Electrónica y Telecomunicaciones, y a todos los catedráticos que me ayudaron a crecer en conocimientos de pregrado.

DEDICATORIA

Br. Cardoza Atoche Víctor Raúl

A Dios por permitirme llegar a este momento tan especial en mi vida; por los triunfos y momentos difíciles que me enseñaron a valorarte cada día más.

A mis padres Raúl y Teresa y mis hermanos quienes han sido los guías en mi camino para llegar a este punto de mi carrera que con su ejemplo, dedicación y palabras de aliento nunca bajaron los brazos para que yo tampoco lo haga, aún cuando todo se complicaba, les dedico todo mi esfuerzo en reconocimiento a todo sacrificio puesto para que yo pueda estudiar, se merecen esto y mucho más.

A mi esposa por su apoyo y animo que me brinda día con día para alcanzar mis metas, tanto profesionales como personales.

A mi hijo Zaid que posiblemente en estos momentos no entienda mis palabras, pero para cuando seas capaz, quiero que te des cuenta de lo que significas para mí. Eres la razón que me levanta cada día a esforzarme por el presente y el mañana porque eres mi principal motivación.

AGRADECIMIENTO

Br. Cajo Sotero Jesús Alberto

Mi agradecimiento se dirige a quien ha moldeado mi camino y me ha guiado por el camino correcto, a Dios, en que todo momento está conmigo para auxiliarme en mis errores y ya no cometerlos otra vez. Él es el quien bendice mi vida.

Gracias a mis Padres por ser los principales artífices de mis sueños, gracias a ellos por confiar, en creer en mí y en mis proyectos. Gracias a mi madre por estar dispuesta a acompañarme cada larga y agotadora noche de estudio, noches que su compañía y la llegada de sus jugos eran como agua en el desierto, gracias a mi Padre por siempre desear y anhelar lo mejor para mi vida.

Agradezco a mis hermanas que son también mis cimientos para la construcción de mi vida profesional, que sentó en mí las bases de responsabilidad y deseos de superación.

Estoy enormemente agradecido a la escuela de Ingeniería Electrónica y Telecomunicaciones que por recursos de sus catedráticos, ingenieros y personal administrativos he gozado de nuevos conocimientos y lecciones que he obtenido.

Gracias a mi maestro Ing. Edwin Ocas Infante, quien estuvo en mis inicios, quien se ha tomado el arduo trabajo de transmitirme sus diversos conocimientos, especialmente del campo y de los temas que corresponden a mi profesión.

Finalmente agradezco a todos las personas que creyeron en mí y en este proyecto.

Br. Cardoza Atoche Víctor Raúl

Gracias a Dios que, con su amor y bondad, está presente no solo en esta etapa tan importante de mi vida, sino en todo momento ofreciéndome lo mejor y buscando lo mejor para mi persona.

Gracias a mis padres ya que, por su amor recibido, la dedicación y la paciencia con la que cada día se preocupaban por mi avance, gracias a ellos por cada día confiar, creer en mí y en mis expectativas; a mi padre por haberme inculcado buenos valores para formar la persona que soy, a mi madre por darme la fuerza y ayudarme a superar cada obstáculo presentado en mi vida.

A mis hermanos que gracias a ellos adquirí el don de la paciencia y la reflexión por compartir alegrías y tropiezos por las cuales hemos salido triunfadores, por su confianza y por permitirme estar en sus vidas.

A mis profesores, compañeros y amigos muchas gracias por el tiempo compartido y por creer en mí.

INDICE

INDICE DE FIGURAS

RESUMEN

En esta tesis se ha desarrollado un prototipo de alimentador automático para animales domésticos utilizando una plataforma en modulo electrónico , con la finalidad de que estos animales sean alimentados correctamente , debidos a que muchas veces la falta de tiempo de sus dueños , por diversos motivos , hacen que estos no coman a su horario correspondiente , perjudicándolos asi en su nutrición, nuestro proyecto de tesis ayudara a solucionar este problema , pues nos permitirá tener un control en el horario y en las cantidades de alimento correspondiente de estos animales domésticos.

Nuestro proyecto de tesis se desarrollara utilizando Arduino uno y la tarjeta de comunicación sim900 , en las cuales se programara el horario de alimentación y el accionamientos de relés que nos permitirá activar el funcionamiento de las válvulas y del motor para el transporte de la comida desde el envase hasta la fuente final .

En el primer capítulo se describe la problemática y se establece el objetivo general y especifico de la tesis, así como la justificaciones importancia, el segundo capítulo corresponde al marco teórico, en el cual se hace mención a los antecedentes y a la teoría referente al proyecto, En el tercer capítulo se muestra la metodología desarrollada durante la investigación, En el cuarto capítulo se muestra el desarrollo del sistema y en el quinto capítulo los resultados y conclusiones

Palabras clave: Gsm, Accionamiento, Domestico, Relés Modulo, Automatico

ABSTRACT

In this thesis a prototype of automatic feeder for domestic animals has been developed using an electronic module platform, so that these animals are fed correctly, because many times the lack of time of their owners, for various reasons, make that these do not eat at their corresponding time, thus damaging them in their nutrition, our thesis project will help to solve this problem, because it will allow us to have a control in the schedule and in the amounts of corresponding food of these domestic animals.

Our thesis project will be developed using Arduino Uno and the sim900 communication card, in which the feeding schedule and relay drives will be programmed that will allow us to activate the operation of the valves and the motor for the transport of food from the container to the final source.

The first chapter describes the problem and establishes the general and specific objective of the thesis, as well as the importance justifications, the second chapter corresponds to the theoretical framework, in which mention is made of the background and theory regarding the project, The third chapter shows the methodology developed during the investigation. The fourth chapter shows the development of the system and the fifth chapter shows the results and conclusions.

Keywords: Gsm, Drive, Domestic, Module Relays, Automatic

INTRODUCCIÓN

Se vive en una era que la tecnología deja de ser una exclusividad de los sectores productivos e invade los hogares de las familias, trayendo más confort, sofisticación y practicidad. Este ambiente tecnológico es cada vez más anhelado por las familias.

De esta forma, los integrantes de esas familias se están volviendo independientes y buscando una carrera profesional. Los animales domésticos, que antes tenían a alguien para cuidar, ahora pasan horas en casa solos. Para que el animal crezca sano es importante que la alimentación sea supervisada y eso requiere cantidades y horarios específicos.

Los animales domésticos terminan siendo mejores compañeros. El afecto y la dedicación para cuidarlos se vuelve primordial en los días de hoy. En este mercado nacional e internacional, cada vez más la demanda de pet shops y clínicas veterinarias están aumentando, consecuentemente promoviendo tipos diferenciados de servicios y un resultado satisfactorio de los clientes (SANTOS: RAMIREZ-GÁLVEZ. P.1).

En la presente tesis se diseñara e implementara un prototipo de alimentador automático para animales domésticos que utiliza una plataforma mediante un módulo electrónico, para así lograr brindar un mejor cuidado y/o atención a los animales domésticos para su correcta alimentación.

1. ASPECTOS DE LA PROBLEMÁTICA

1.1. Descripción de la realidad problemática

Hoy en día los animales de compañía vienen presentando un gradualmente aumento de peso debido a la mala alimentación. El origen de este sobrepeso generalmente es causado por la falta de control de la cantidad de alimento servido al animal. Normalmente está disponible durante todo el día, cuando debería ser servido en horas y cantidad controladas. Esta falta de control puede estar asociada a factores como falta de tiempo de los dueños, que pueden no estar presentes cuando es necesario u olvidarse de los horarios, e incluso algún viaje, donde el animal no puede estar junto y debe quedarse en casa solo.

En esta investigación se pretende diseñar e implementar un prototipo de alimentador automático para animales domésticos utilizando una plataforma mediante un módulo electrónico.

1.2. Definición del problema

La problemática radica en que no se recomienda que un animal doméstico tenga acceso a una gran cantidad de alimento en una sola comida y no es posible realizar ese control sin un sistema automatizado.

Proporcionar toda la ración en una sola vez da libertad al perro en decidir cuánto y cuándo comer, dificultando la correcta alimentación y nutrición del animal doméstico.

1.3. Formulación del problema de investigación

¿Sera posible diseñar e implementar un prototipo de alimentador automático para animales domésticos que utilice una plataforma mediante un módulo electrónico?

1.4. Justificación e importancia de la investigación

1.4.1. Justificación de la investigación

El desarrollo de la investigación nace por la necesidad de tener un sistema automatico que permita la alimentación de los animales domésticos de forma gradual, pues es esencial el cuidado con la salud de los animales. Un alimentador automatizado haría que las dificultades encontradas sean sanadas.

1.4.2. Importancia de la investigación

Esta investigación es muy importante ya que solucionara las necesidades que se presentan en la crianza de los animales (alimentación) la cual ayudara tanto al dueño como al animal, logrando que su alimentación sea en el tiempo correspondiente y de forma saludable.

1.5. Objetivos

1.5.1. Objetivo general

Diseñar un prototipo de alimentador automático para animales domésticos que ayude en su correcta alimentación en horarios y en proporciones racionadas utilizando un prototipo electrónico.

1.5.2. Objetivos específicos

- Realizar y Aplicar conocimiento en electrónica desarrollando un prototipo electrónico para la alimentación automática de animales domésticos
- Aplicar conocimientos mecánicos en la elaboración de una estructura estable y resistente.
- Diseñar el software para que cumpla las funciones requeridas
- Interactuar con nuestro prototipo mediante la tecnología GSM/GPRS

II. MARCO TEÓRICO

2. Antecedentes de la investigación

En la provincia de Piura se han registrado proyectos que buscan mejorar la alimentación de los animales, pero no existe antecedentes de la elaboración de un alimentador automático de animales domésticos, sin embargo en la actualidad existen alguna investigación y proyectos desarrollados en diversos países del mundo tales como:

EN LA DISERTACION TITULADA " PROYECTO DE FACTIBILIDAD PARA LA CREACION DE UN DISPENSADOR DE ALIMENTOS AUTOMATIZADO PARA PERROS EN EL DISTRITO METROPOLITANO DE QUITO" en el año 2012 por Guido Antonio Acosta Laso, el cual es un estudio de la viabilidad de un dispensador automatico, donde se concluye que el dispensador de alimentos son productos nuevos y cuentan con un mercado amplio, y según las encuestas realizadas, 4739 personas en quito, estarían dispuestas adquirir un dispensador de comida.

En la tesis titulada "DESARROLLO DE UN SISTEMA DE DOSIFICACION AUTOMATICO DE ALIMENTOS PARA EQUINOS"

En el año 2010 por Franco Guzmán, Luis Fernando, Galicia Jimenez, Jesus Rodolfo y Ostria Valle Diana. Se plantea desarrollar un sistema dosificador automatico de alimento para equinos de alto rendimiento mediante la implementación de un sistema de control electro neumático que automatice el proceso de alimentación manual que se sigue en los centros hípicos de Mexico. Con el fin de proporcionar una dosis que se adecue a las necesidades particulares de ingesta diaria de grano que requieren estos animales.

2.1. Bases teóricas

2.1.1. **Animales Domésticos**

Los animales domésticos son pequeños o grandes animales que pueden llegar a ser domesticados por el hombre y, por tanto, convivir con ellos. Cuando pensamos en animales domésticos lo hacemos en perros, gatos, etc., pero también lo son los caballos, las gallinas, etc. porque son animales domesticados por el hombre. Los animales de la granja también son considerados como domésticos.

Dentro de los animales domésticos encontramos los animales de compañía, que son los que los humanos tienen en casa. También son llamados mascotas. Ya sea un gato, un perro, un pájaro o los roedores, las mascotas se convierten casi en miembros de la familia.

Los animales de compañía pueden tener una función útil como el perro guardián o el gato que caza a los roedores, pero hay otros que los tenemos sólo como compañía o entretenimiento, como por ejemplo los pájaros, que nos ofrecen sus dulces y armoniosos cantos. Normalmente, los animales domésticos poseen un efecto positivo en la gente, ya que los miramos actuar con atención y curiosidad.

Los gatos y los perros son animales afectivos a los que les gustan los mimos, las atenciones y jugar. Son los animales más cercanos a la familia y se convierten en un miembro más de la misma. Los roedores, las tortugas y los lagartos son animales más fríos. Son animales de compañía, pero que trasmiten menos sensaciones y que, sobre todo, necesitan menos contacto con el ser humano.

Los peces no se tienen por su utilidad, pero sí por estética ya que existen multitud de peces con multitud de formas y colores totalmente variados. Además, no son animales ruidosos y necesitan pocos cuidados (dependiendo del tipo de pez y de la cantidad que se tenga).

Los pájaros domésticos son muy populares por sus cantos y sus variados y vivos colores. Pero debemos aclarar que el canto de un pájaro puede ser muy agradable o muy cansado, depende del tipo de persona que seas. Los insectos raramente son utilizados como animal de compañía. Aunque sí hay a quien le gustan las grandes arañas y las mantienen en un acuario.

Tu animal doméstico debe tener una buena alimentación para estar saludable y ser feliz. Existen productos alimenticios especiales y recomendados para cada tipo de animal de compañía. (https://www.anipedia.net/mundo-animal/animales-domesticos/)

2.1.1.1. **Animales domésticos más comunes**

A continuación se presenta una lista de los más frecuentes animales domésticos:

1. **Perro** (Canis lupus familiaris).	21. **Ratón casero** (Mus musculus)
2. **Gallo** (Gallus gallus)	22. **Tórtola rosigris** (Sreptopelia roseogrisea)
3. **Gato** (Felis silvestris catus)	23. **Pavo** (Meleagris gallopavo)
4. **Vaca** (Bos primigenius taurus)	24. **Pez carpa** (Cyprinus carpio)
5. **Toro cebú** (Bos primigenius indicus)	25. **Rata doméstica** (Rattus norvegicus)
6. **Cabra** (Capra aegagrus hircus)	26. **Canario doméstico** (Serinus canaria domestica)
7. **Cerdo** (Sus scrofa domestica)	27. **Pez guppy o pez millón** (Poecilia reticulata)
8. **Oveja** (Ovis orientalis aries)	28. **Abeja doméstica** (Apis mellifera)
9. **Cuy** (Cavia porcellus)	29. **Pato criollo** (Cairina moschata)
10. **Burro** (Equus africanus asinus)	30. **Pavo real** (Pavo cristatus)
11. **Pato doméstico** (Anas platyrhynchos domesticus)	31. **Cacatúa** (Cacatua galerita)
12. **Caballo** (Equus ferus)	32. **Guacamaya** (Ara macao)
13. **Dromedario** (Camelus dromedarius)	33. **Tortuga de tierra** (Chelonoidis carbonaria)
14. **Gusano de seda** (Bombyx mori)	34. **Cisne** (Cygnus olor)
15. **Paloma común** (Columba livia domestica)	35. **Periquito australiano** (Melopsittacus undulatus)
16. **Camello** (Camelus bactrianus)	36. **Mosca de la fruta** (Drosophila melanogaster)
17. **Llama** (Llama glama)	37. **Capibara, chigüire o carpincho** (Hydrochoerus hydrochaeris)
18. **Alpaca** (Vicugna pacos)	38. **Hámster** (Mesocricetus auratus)
19. **Gallina de Guinea** (Numida meleagris)	39. **Tortuga de orejas rojas** (Trachemys scripta elegans)
20. **Turón** (Mustela putorius)	40. **Loro doméstico** (Psittacidae spp.)

Tabla 01: animales domésticos comunes.

Fuente: https://concepto.de/animales-domesticos

2.1.1.1.1. El perro

Llamamos perro a un conjunto de especies de cánidos domésticos, emparentados con el lobo salvaje, que hace alrededor de 10.000 años emprendió un modo de vida próximo al ser humano, probablemente entendiendo que una asociación con nuestra especie podía resultarle benéfico en términos de fácil acceso a la comida, calor y techo, a cambio de protección y asistencia en la cacería.

El paso de los siglos, no obstante, terminó haciendo del perro un animal de compañía con enorme variedad entre una raza y la otra, debido en parte a nuestra intromisión mediante el cruce selectivo.(https://concepto.de/animales-domesticos/#ixzz65l10qMh4)

Figura 1: Imagen de animal doméstico (perro)

Fuente:.(https://concepto.de/animales-domesticos)

2.1.1.1.2. El gato

Otro de los animales de compañía más comunes es el gato, aunque de una domesticación menos completa, en apariencia, que la del perro, dado que conserva una buena parte de sus instintos de cacería intactos. Se piensa que se lo introdujo a la civilización humana como una forma de dar cacería a los roedores que infestaban los depósitos de alimento de la civilización del Antiguo Egipto.

Este tipo de felino de mediano tamaño fue venerado por numerosas civilizaciones orientales y condenado por el cristianismo occidental, que vio en ellos un símbolo del mal, seguramente debido a sus hábitos nocturnos e independientes.(https://concepto.de/animales-domesticos/#ixzz65l1BGy5s)

Figura 2: Imagen de animal doméstico (gato)

Fuente:.(https://concepto.de/animales-domesticos)

2.1.1.1.3. El loro

Una de las aves domésticas más sociables conocidas, de plumaje típicamente verde aunque con otros colores accesorios, el loro destaca por su fuerte pico curvo y su capacidad para imitar el lenguaje humano. No se trata, sin embargo, de que el loro realmente “hable”, dado que es incapaz de adquirir el lenguaje; pero sí es capaz de imitar bastante fielmente diversas palabras, así como de hacer otros sonidos como silbidos, risas, etc.(https://concepto.de/animales-domesticos/#ixzz65l1L1K1s)

Figura 3: Imagen de animal doméstico (loro)

Fuente:.(https://concepto.de/animales-domesticos)

2.1.1.1.4. La vaca

Quizá el animal doméstico más importante para la historia humana sea la vaca, o al menos los bovinos en general. No sólo porque de ella obtenemos leche, fuente de numerosos productos alimenticios, y diversos tipos de carnes o de cueros, para alimentarnos o para protegernos del frío; sino también porque su introducción a las primitivas civilizaciones humanas permitió que el arado se llevara a cabo de manera mucho más eficaz, aprovechando la fuerza bruta del animal para abrir surcos en la tierra y poder luego sembrar.

(https://concepto.de/animales-domesticos/#ixzz65l1UkwUe)

Figura 4: Imagen de animal doméstico (vaca)

Fuente:.(https://concepto.de/animales-domesticos)

2.1.1.1.5. El caballo

Otro de los animales domésticos más significativos de nuestra historia, asociado a la fuerza, la velocidad y el espíritu salvaje, sirvió de transporte al humano durante milenios, ya fuera montándolo directamente o usándolo como tracción de diversos vehículos a ruedas (carretas, carrozas, etc). A tal grado llega la domesticación del caballo que solemos intervenir directamente su cuerpo, clavándole herraduras metálicas en las pezuñas para protegerlas del desgaste, o retirándoles dientes para insertar las bridas. (https://concepto.de/animales-domesticos/#ixzz65l1c6yLR)

Figura 5: Imagen de animal doméstico (caballo)

Fuente:.(https://concepto.de/animales-domesticos)

2.1.1.2. **Alimentación de los animales domésticos**

Para los animales domésticos existe una alimentación más limitada que para los animales salvajes, esto se debe al control que tienen los dueños de los alimentos que consumen sus mascotas.

Esto no siempre significa que todas las mascotas tengan dueños responsables, cada una de las especies y razas necesitan y merecen del alimento adecuado para su desarrollo.

Los perros y gatos deben comer los alimentos elaborados específicamente para ellos, existen alimentos de consumo humano como el chocolate, la pasta, el arroz y enlatados que pueden ser mortales para el animal.

En cierto aspecto son vulnerables porque ellos no son capaces de obtener sus alimentos por sus propios medios, también debes tomar en cuenta que el animal no tiene criterio de selección, él confía en ti y en los alimentos que le das, por ende procura que sean los más saludables para su bienestar.

Si tienes o tendrás animales como caballos, ardillas, aves, cabras y demás animales bajo tú responsabilidad, procura alimentarlos de la mejor manera posible.

Los periquitos, loros y demás aves necesitan de sumo cuidado, estos animales son muy sensibles y sentimentales, pueden desarrollar una estrecha relación con sus cuidadores, así que el cariño y amor de parte de ellos es fundamental para la salud del ave. Estos animales domésticos son muy buena compañía para personas de la tercera edad como para los niños, que junto a las aves desarrollan buenos valores y comportamientos. (https://animalesis.com/domesticos/).

Figura 6: Imagen de animal doméstico (caballo)

Fuente:.(https://animalesis.com/domesticos)

2.1.2. **ARDUINO UNO R3**

Arduino Uno es una placa electrónica basada en el microcontrolador ATmega328. Cuenta con 14 entradas/salidas digitales, de las cuales 6 se pueden utilizar como salidas PWM (Modulación por ancho de pulsos) y otras 6 son entradas analógicas. Además, incluye un resonador cerámico de 16 MHz, un conector USB, un conector de alimentación, una cabecera ICSP y un botón de reseteado. La placa incluye todo lo necesario para que el microcontrolador haga su trabajo, basta conectarla a un ordenador con un cable USB o a la corriente eléctrica a través de un transformador.

Figura 7: placa electrónica Arduino uno r3

Fuente: IEEE Argentina, Miguel Lattanzi

2.1.2.1. **Características técnicas de Arduino Uno r3**

- Microcontrolador: ATmega328
- Voltage: 5V
- Voltage entrada (recomendado): 7-12V
- Voltage entrada (limites): 6-20V
- Digital I/O Pins: 14 (de los cuales 6 son salida PWM)
- Entradas Analogicas: 6
- DC Current per I/O Pin: 40 mA
- DC Current parar 3.3V Pin: 50 mA
- Flash Memory: 32 KB (ATmega328) de los cuales 0.5 KB son utilizados para el arranque
- SRAM: 2 KB (ATmega328)
- EEPROM: 1 KB (ATmega328)
- Clock Speed: 16 MHz

2.1.2.2. **Mapeo de la placa Arduino**

Mirando a la placa desde la parte de arriba, este es el esquema de lo que puedes ver (los componentes de la placa con los que puedes interactuar en su uso normal están resaltados):

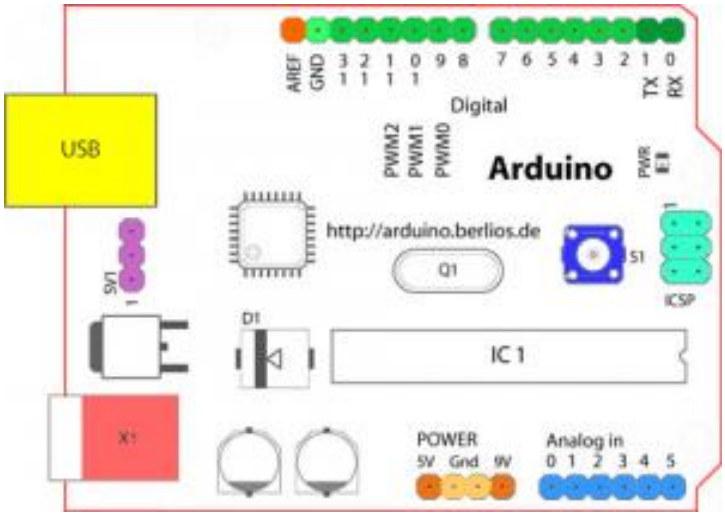


Figura 8: pines Arduino uno r3

Fuente: www.iescamp.es

Empezando según las agujas del reloj:

- Terminal de referencia analógica (naranja).
- Tierra digital (verde claro)
- Terminales digitales 2-13 (verde)
- Terminales digitales 0-1/ E/S serie – TX/RX (verde oscuro) – Estos pines no se pueden utilizar como e/s digitales (digitalRead() y digitalWrite()) si estás utilizando comunicación serie .
- Botón de reinicio – S1 (azul oscuro)
- Terminales de entrada analógica 0-5 (azul claro)
- Terminales de alimentación y tierra (alimentación: naranja, tierras: naranja claro)
- Entrada de alimentación externa (9-12VDC) – X1 (rosa)

- Selector de alimentación externa o por USB (coloca un jumper en los dos pines mas cercanos de la alimentación que quieras)
- USB (utilizado para subir programas a la placa y para comunicaciones serie entre la placa y el ordenador; puede utilizarse como alimentación de la placa) (amarillo)

2.1.2.2.1. Entradas y salidas digitales/analógicas

Un sistema electrónico es cualquier disposición de componentes electrónicos con un conjunto definido de entradas y salidas. Una placa Arduino, por tanto, puede pensarse de forma simplificada como un sistema que acepta información en forma de señal de entrada, desarrolla ciertas operaciones sobre ésta y luego produce señales de salida.

Justamente, una de las opciones que hacen más potente a Arduino son sus entradas/salidas digitales. ¿Entonces por qué hablamos de analógicas?

En los sistemas electrónicos, una magnitud física variable se representa generalmente mediante una señal eléctrica que varía de manera tal que describe esa magnitud. Por lo general, se hace referencia a las señales continuas como señales analógicas, mientras que asociamos las señales discretas a señales digitales: el ejemplo más claro es el de las señales binarias, donde la señal sólo pueden tomar dos niveles, 0 o 1.

Arduino incorpora terminales digitales (señales discretas) pero de tal forma que tenemos un gran abanico de valores con los que trabajar (por ejemplo, 255 valores de luz en un fotosensor, siendo 0 ausencia de luz y 254 el máximo valor lumínico).

2.1.2.2.2. Terminales Digitales

Las terminales digitales de una placa Arduino pueden ser utilizadas para entradas o salidas de propósito general a través de los comandos de programación, pinMode(), digitalRead(), y digitalWrite().

Cada terminal tiene una resistencia pull-up que puede activarse o desactivarse utilizando digitalWrite() (con un valor de HIGH o LOW, respectivamente) cuando el pin esta configurado como entrada. La corriente máxima por salida es 40 mA.

- **Serial:** 0 (RX) y 1 (TX). Utilizado para recibir (RX) y transmitir (TX) datos serie TTL. En el Arduino Diacemila, estas terminales están conectadas a las correspondientes patas del circuito integrado conversor FTDI USB a TTL serie. En el Arduino BT, están conectados al las terminales correspondientes del modulo Bluetooth WT11. En el Arduino Mini y el Arduino LilyPad, están destinados para el uso de un módulo serie TTL externo (por ejemplo el adaptador Mini-USB).
- **Interruptores externos**: 2 y 3. Estas terminales pueden ser configuradas para disparar una interrupción con un valor bajo, un pulso de subida o bajada, o un cambio de valor. Mira la función attachInterrupt() para mas detalles.
- **PWM**: 3, 5, 6, 9, 10, y 11. Proporcionan salidas PWM de 8 bit con la función analogWrite(). En placas con ATmega8, las salidas PWM solo están disponibles en los pines 9, 10, y 11.
- **Rese**t BT: 7. (solo en Arduino BT) Conectado a la línea de reset del módulo bluetooth.
- **SPI:** 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). Estas terminales soportan comunicación SPI. Aunque esta funcionalidad esta proporcionada por el hardware, no está incluida actualmente el el lenguaje Arduino.

- **LED: 13**. En el Diacemila y el LilyPad hay un led en placa conectado al pin digital 13. cuando el pin tiene valor HIGH, el LED está encendido, cuando el pin está en LOW, está apagado

- **Pines Analógicos:** Los pines de entrada analógicos soportan conversiones analógico-digital (ADC) de 10 bit utilizando la función analogRead(). Las entradas analógicas pueden ser también usadas como pines digitales: entrada analógica 0 como pin digital 14 hasta la entrada analógica 5 como pin digital 19. Las entradas analógicas 6 y 7 (presentes en el Mini y el BT) no pueden ser utilizadas como pines digitales.

- **I2C**: 4 (SDA) y 5 (SCL). Soportan comunicaciones I2C (TWI) utilizando la librería Wire (documentación en la página web de Wiring).

Pines de alimentación

- **VIN** (a veces marcada como «9V»). Es el voltaje de entrada a la placa Arduino cuando se está utilizando una fuente de alimentación externa (En comparación con los 5 voltios de la conexión USB o de otra fuente de alimentación regulada). Puedes proporcionar voltaje a través de este pin. Date cuenta que diferentes placas aceptan distintos rangos de voltaje de entrada, por favor, mira la documentación de tu placa. También date cuenta que el LilyPad no tiene pin VIN y acepta solo una entrada regulada.

- **5V**. La alimentación regulada utilizada para alimentar el microcontrolador y otros componentes de la placa. Esta puede venir de VIN a través de un regulador en placa o ser proporcionada por USB u otra fuente regulada de 5V.

- **3V3.** Una fuente de 3.3 voltios generada por el chip FTDI de la placa.

- **GND**. Pines de tierra.

Otros Pines

- **AREF.** Referencia de voltaje para las entradas analógicas. Utilizada con la función analogReference().
- **Reset.** Pon esta línea a LOW para resetear el microcontrolador. Utilizada típicamente para añadir un botón de reset a shields que bloquean el de la placa principal.

  (http://www.iescamp.es/miarduino/2016/01/21/placa-arduino-uno/)

2.1.3. **RED GPRS/GSM**

2.1.3.1. **Estándar GSM**

La red GSM (Sistema global de comunicaciones móviles) es, a comienzos del siglo XXI, el estándar más usado en Europa. Se denomina estándar de segunda generación (2G) porque, a diferencia de la primera generación de teléfonos portátiles, las comunicaciones se producen de un modo completamente digital.

En 1982, cuando fue estandarizado por primera vez, fue denominado Groupe Spécial Mobile y en 1991 se convirtió en un estándar internacional llamado Sistema Global de Comunicaciones Móviles.

En Europa, el estándar GSM usa las bandas de frecuencia de 900 MHz y 1.800 MHz. Sin embargo, en los Estados Unidos se usa la banda de frecuencia de 1.900 MHz. Por esa razón, los teléfonos portátiles que funcionan tanto en Europa como en los Estados Unidos se llaman tribanda y aquellos que funcionan solo en Europa se denominan bibanda.

El estándar GSM permite un rendimiento máximo de 9,6 kbps, que permite transmisiones de voz y de datos digitales de volumen bajo, por ejemplo, mensajes de texto (SMS, Servicio de mensajes cortos) o mensajes multimedia (MMS, Servicio de mensajes multimedia).

2.1.3.1.1. Red celular

Las redes de telefonía móvil se basan en el concepto de celdas, es decir zonas circulares que se superponen para cubrir un área geográfica.

Las redes celulares se basan en el uso de un transmisor-receptor central en cada celda, denominado estación base (o Estación base transceptora, BTS). Cuanto menor sea el radio de una celda, mayor será el ancho de banda disponible. Por lo tanto, en zonas urbanas muy pobladas, hay celdas con un radio de unos cientos de metros mientras que en zonas rurales hay celdas enormes de hasta 30 kilómetros que proporcionan cobertura.

En una red celular, cada celda está rodeada por 6 celdas contiguas (por esto las celdas generalmente se dibujan como un hexágono). Para evitar interferencia, las celdas adyacentes no pueden usar la misma frecuencia. En la práctica, dos celdas que usan el mismo rango de frecuencia deben estar separadas por una distancia equivalente a dos o tres veces el diámetro de la celda.

2.1.3.1.2. Arquitectura de la red GSM

En una red GSM, el terminal del usuario se llama estación móvil. Una estación móvil está constituida por una tarjeta SIM (Módulo de identificación de abonado), que permite identificar de manera única al usuario y al terminal móvil, o sea, al dispositivo del usuario (normalmente un teléfono portátil).

Los terminales (dispositivos) se identifican por medio de un número único de identificación de 15 dígitos denominado IMEI (Identificador internacional de equipos móviles). Cada tarjeta SIM posee un número de identificación único (y secreto) denominado IMSI (Identificador internacional de abonados móviles). Este código se puede proteger con una clave de 4 dígitos llamada código PIN.

Por lo tanto, la tarjeta SIM permite identificar a cada usuario independientemente del terminal utilizada durante la comunicación con la estación base. Las comunicaciones entre una estación móvil y una estación base se producen a través de un vínculo de radio, por lo general denominado interfaz de aire (o en raras ocasiones, interfaz Um).

Todas las estaciones base de una red celular están conectadas a un controlador de estaciones base (o BSC), que administra la distribución de los recursos. El sistema compuesto del controlador de estaciones base y sus estaciones base conectadas es el Subsistema de estaciones base (o BSS).

Por último, los controladores de estaciones base están físicamente conectados al Centro de conmutación móvil (MSC) que los conecta con la red de telefonía pública y con Internet; lo administra el operador de la red telefónica. El MSC pertenece a un Subsistema de conmutación de red (NSS) que gestiona las identidades de los usuarios, su ubicación y el establecimiento de comunicaciones con otros usuarios.

Generalmente, el MSC se conecta a bases de datos que proporcionan funciones adicionales, como el Registro de ubicación de origen (HLR), una base de datos que contiene información (posición geográfica, información administrativa, etc.) de los abonados registrados dentro de la zona del conmutador (MSC); el Registro de ubicación de visitante (VLR), una base de datos que contiene información de usuarios que no son abonados locales. El VLR recupera los datos de un usuario nuevo del HLR de la zona de abonado del usuario. Los datos se conservan mientras el usuario está dentro de la zona y se eliminan en cuanto abandona la zona o después de un período de inactividad prolongado (terminal apagado); el Registro de identificación del equipo (EIR), una base de datos que contiene la lista de terminales móviles; el Centro de autenticación (AUC), que verifica las identidades de los usuarios.

La red celular compuesta de esta manera está diseñada para admitir movilidad a través de la gestión de traspasos (movimientos que se realizan de una celda a otra).

Finalmente, las redes GSM admiten el concepto de roaming: el movimiento desde la red de un operador a otra. (https://es.ccm.net/contents/681-estandar-gsm-sistema-global-de-comunicaciones-moviles)

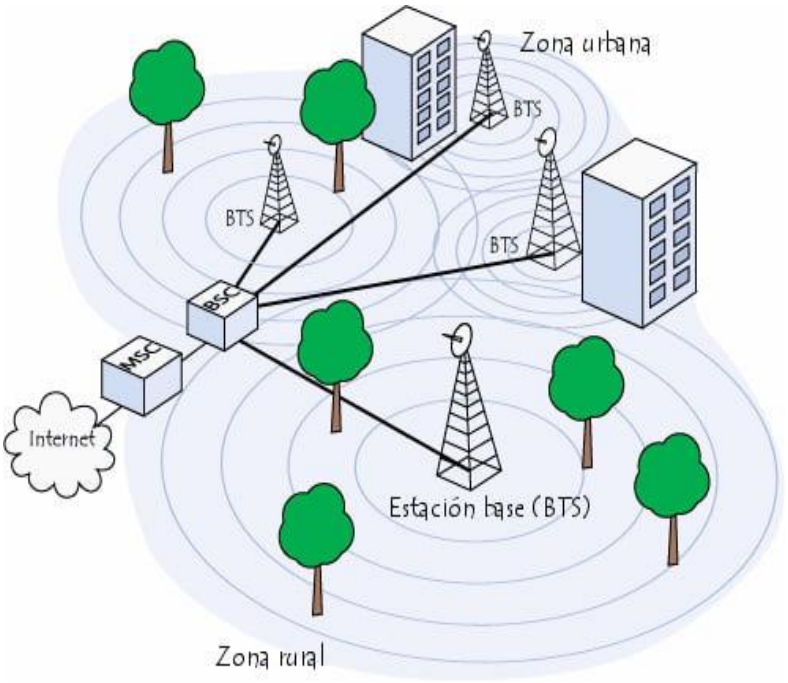


Figura 9: redes GSM

Fuente:es.ccm.net/contents/681-estandar-gsm-sistema-global-de-comunicaciones-moviles

2.1.3.1.3. Tarjeta SIM

Una tarjeta SIM contiene la siguiente información: el número telefónico del abonado (MSISDN); el número internacional de abonado (IMSI, Identificación internacional de abonados móviles), el estado de la tarjeta SIM; el código de servicio (operador); la clave de autenticación; el PIN

(Código de identificación personal); el PUK (Código personal de desbloqueo).

(https://es.ccm.net/contents/681-estandar-gsm-sistema-global-de-comunicaciones-moviles)

2.1.3.2. **Estándar GPRS**

El estándar GPRS (General Packet Radio Service) es una evolución del estándar GSM y es por eso que en algunos casos se denomina GSM++ (o GMS 2+). Dado que es un estándar de telefonía de segunda generación que permite una transición hacia la tercera generación (3G), el estándar GPRS por lo general se clasifica como 2.5G.

GPRS extiende la arquitectura del estándar GSM para permitir la transferencia de datos del paquete con una tasa de datos teóricos de alrededor de 171,2 Kbits/s (hasta 114 Kbits/s en la práctica). Gracias a su modo de transferencia en paquetes, las transmisiones de datos sólo usan la red cuando es necesario. Por lo tanto, el estándar GPRS permite que el usuario reciba facturas por volumen de datos en lugar de la duración de la conexión, lo que significa especialmente que el usuario puede permanecer conectado sin costo adicional.

Para el transporte de voz, el estándar GPRS emplea la arquitectura de red GSM y provee acceso a la red de datos (especialmente Internet) por medio del protocolo IP o del protocolo X.25.

GPRS admite características nuevas que no están disponibles en el estándar GSM y que se pueden clasificar en los siguientes tipos de servicios:

- Servicio de punto a punto (PTP): es la capacidad de conectarse en modo cliente-servidor a un equipo en una red IP.

- Servicio de punto a multipunto (PTMP): constituye la capacidad de enviar paquetes a un grupo de destinatarios (Multidifusión).
- Servicio de mensajes cortos (SMS)

2.1.3.2.1. Arquitectura de la red GPRS

La integración de GPRS a una arquitectura GSM requiere que se añadan nuevos nodos de red denominados GSN (nodos de soporte GPRS) ubicados en una red de transporte:

- el router SGSN (Nodo de soporte de servicio GPRS) gestiona las direcciones de las terminales de la celda y proporciona la transferencia de la interfaz de paquetes con la pasarela GGSN.
- la pasarela GGSN (Nodo de soporte de pasarela GPRS) se conecta con otras redes de datos (Internet). En particular, GGSN debe proporcionar una dirección IP a las terminales móviles durante toda la conexión.

2.1.3.3. GSM / GPRS - Arduino Shield SIM900

El shield GPRS se basa en el módulo SIM900 de SIMCOM y compatible con Arduino y sus clones. El shield GPRS le proporciona una manera de comunicarse mediante la red de telefonía celular GSM. El escudo le permite lograr SMS, MMS, GPRS y audio a través de UART mediante el envío de comandos AT (GSM 07.07, 07.05 y SIMCOM realzada Comandos AT). El shield también tiene los 12 GPIO, 2 PWM y un ADC del módulo SIM900 (Son toda lógica 2V8) presentes a bordo. (https://www.carrod.mx/products/modulo-gsm-gprs-sim900)

Características

- Voltaje de alimentación mínima: 9 V
- Voltaje de alimentación mínima: 20 V
- Corriente: 1.5 mA (modo descanso)
- Comunicación: UART
- Bandas de frecuencia: 850/900/1800/1900MHz
- Multi-GPRS Slot: clase 10/8
- Estación móvil: GPRS Clase Base: GSM 2/2+
- Clase 1 1W a: 1800/1900 MHz
- Clase 4 2W a: 850/900 MHz
- Servicio de mensajería (cortos): Envío de pequeñas cantidades de datos a través de la red (ascii o primas hexadecimal)
- Pila embebida TCP/UDP: Carga de datos a un servidor web
- Puerto serie: Libre selección
- Portabatería para batería: CR1220 (no incluida)
- Altavoz y tomas de auriculares: 2 conectores Jack 3.5 mm
- Temperatura de operación mínima: -40 °C
- Temperatura de operación máxima: 85 °C
- Soporta: SuperCap
- Dimensiones: 75 mm X 55 mm X 10 mm
- Modelo: SIM900

2.1.3.3.1. Precauciones

- Asegúrese de que su tarjeta SIM está desbloqueado.
- El producto se proporciona tal cual y sin un recinto aislante. Tenga en cuenta las precauciones ESD especialmente en (baja humedad) seco clima.
- La configuración predeterminada de fábrica para el GPRS Escudo UART es 19200 bps 8-N-1. (Se puede cambiar mediante comandos AT).

Figura 10: shiel Arduino GSM/GPRS SIM900

Fuente: https://es.scribd.com/sim900

2.1.3.3.2. Hardware Diagrama

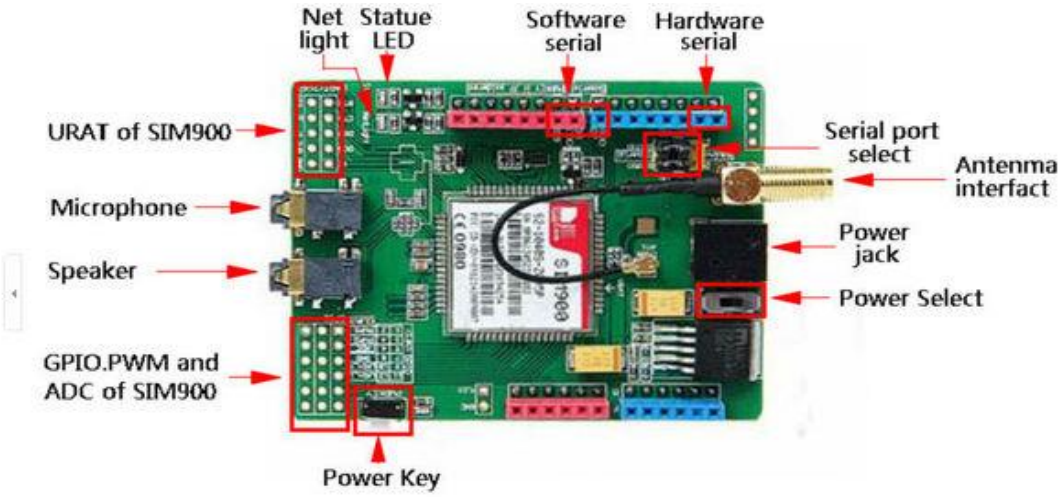


Figura 11: shiel Arduino GSM/GPRS SIM900

Fuente: wiki.seeedstudio.com/GPRS_Shield_V2.0/

Poder seleccionar - seleccione la fuente de alimentación para el escudo GPRS (poder o 5v de arduino externa)

- Toma de alimentación - conectado a la fuente de alimentación externa 4,8 ~ 5 V CC
- Interfaz de la antena - conectado a la antena externa
- Puerto serie seleccione - seleccionar cualquiera de los puertos serie del software o un puerto serie hardware estar conectado a GPRS shield
- Serial Hardware - D0 / D1 de Arduino
- serial Software - D7 / D8 de Arduino
- La luz neta - dirá el estado acerca SIM900 ligarse a la red
- UART de SIM900 - pines UART ruptura de SIM900
- Micrófono - para responder a la llamada de teléfono
- Altavoz - para responder a la llamada de teléfono
- GPIO, PWM y ADC de SIM900 - GPIO, pines PWM y ADC ruptura de SIM900
- Tecla de encendido - el poder arriba y hacia abajo para SIM900
- usa Pines, en Arduino
- D7 - si se selecciona el puerto serie de software para comunicarse con GPRS Escudo
- D8 - si se selecciona el puerto serie de software para comunicarse con GPRS Escudo
- D9 - Se utiliza para el control de software de la energía para arriba o abajo de la SIM900

2.1.3.3.3. Comandos at sim900

Para programar el módulo sim900 se requerirá utilizar comandos at, comandos propios que los da el fabricante, si bien es cierto la lista de comandos es muy amplia, para realizar nuestro proyecto se utilizaran comandos principales para configurar las sim900, a continuación se muestra los comandos AT más utilizados

- **AT :**Sirve para verificar si el módulo SIM900 está funcionando adecuadamente para entrar en modo comando. Al enviar AT el SIM deberá contestarnos con un OK.
- **AT+CGMI:** Veremos en nombre del fabricante
- **ATI:**Ver la información del producto.
- **AT+IPR=?** :Preguntar el Baud Rate en el que puede operar el SIM
- **AT+IPR?** :Sirve para preguntar el Baud Rate actual
- **AT+IPR=XXXX :**Configuremos a la frecuencia deseada
- **AT+COPS? :**Nombre de la compañía telefónica
- **AT+CGSN :**Visualizar el IMEI del chip utilizado
- **AT+CSCS?:** Tipo de texto
- **AT+CSCS="XXX"** :Configurar a tipo de texto
- **AT+CMGF? :**Ver el formato de un mensaje, ya sea PDU(0) o SMS(1)"
- **AT+CMGS=04455XXXXXXXX:** Enviar un SMS Se despliega el símbolo mayor que > Escribir mensaje y al finalizar presiona Ctrl+Z retornará OK si el SMS se envió correctamente.
- **AT+CMGL=ALL** :Sirve para ver todos los mensajes que nos han llegado al SIM

- **ATD04455XXXXXXXX:** Sirve para hacer una llamada a cualquier teléfono móvil ATA Sirve para contestar una llamada ATH Sirve para colgar una llamada

2.1.4. **Actuadores y periféricos de salida**

Un actuador es un dispositivo capaz de transformar energía hidráulica, neumática o eléctrica en la activación de un proceso con la finalidad de generar un efecto sobre elemento externo. Este recibe la orden de un regulador, controlador o en nuestro caso un Arduino y en función a ella genera la orden para activar un elemento final de control como, por ejemplo, una válvula.

Existen varios tipos de actuadores como son:

- Electrónicos
- Hidráulicos
- Neumáticos
- Eléctricos

Los actuadores hidráulicos, neumáticos y eléctricos son usados para manejar aparatos mecatrónicos. Por lo general, los actuadores hidráulicos se emplean cuando lo que se necesita es potencia, y los neumáticos son simples posicionamientos. Sin embargo, los hidráulicos requieren mucho equipo para suministro de energía, así como de mantenimiento periódico. Por otro lado, las aplicaciones de los modelos neumáticos también son limitadas desde el punto de vista de precisión y mantenimiento.

2.1.4.1. **Actuadores electrónicos**

Los actuadores electrónicos también son muy utilizados en los aparatos mecatrónicos, como por ejemplo, en los robots. Los servomotores CA sin escobillas se utilizarán en el futuro como actuadores de posicionamiento preciso debido a la demanda de funcionamiento sin tantas horas de mantenimiento.

2.1.4.2. **Actuadores hidráulicos**

Los actuadores hidráulicos, que son los de mayor antigüedad, pueden ser clasificados de acuerdo con la forma de operación, funcionan sobre la base de fluidos a presión. Existen tres grandes grupos:

- cilindro hidráulico
- motor hidráulico
- motor hidráulico de oscilación

2.1.4.2.1. Cilindro hidráulico

De acuerdo con su función podemos clasificar a los cilindros hidráulicos en dos tipos: de efecto simple y de acción doble. En el primer tipo se utiliza fuerza hidráulica para empujar y una fuerza externa, diferente, para contraer. El segundo tipo se emplea la fuerza hidráulica para efectuar ambas acciones. El control de dirección se lleva a cabo mediante un solenoide. En el interior poseen un resorte que cambia su constante elástica con el paso de la corriente. Es decir, si circula corriente por el pistón eléctrico este puede ser extendido fácilmente.

2.1.4.2.2. Cilindro de presión dinámica

Lleva la carga en la base del cilindro. Los costos de fabricación por lo general son bajos ya que no hay partes que resbalen dentro del cilindro.

2.1.4.2.3. Cilindro de simple efecto

La barra esta solo en uno de los extremos del pistón, el cual se contrae mediante resortes o por la misma gravedad. La carga puede colocarse solo en un extremo del cilindro.

2.1.4.2.4. Cilindro de doble efecto

La carga puede colocarse en cualquiera de los lados del cilindro. Se genera un impulso horizontal debido a la diferencia de presión entre los extremos del pistón.

2.1.4.2.5. Cilindro telescópico

La barra de tipo tubo multietápico es empujada sucesivamente conforme se va aplicando al cilindro aceite a presión. Se puede lograr una carrera relativamente larga en comparación con la longitud del cilindro.

2.1.4.2.6. Motor hidráulico

En los motores hidráulicos, el movimiento rotatorio es generado por la presión. Estos motores los podemos clasificar en dos grandes grupos: El primero es uno de tipo rotatorio en el que los engranajes son accionados directamente por aceite a presión, y el segundo, de tipo oscilante, el movimiento rotatorio es generado por la acción oscilatoria de un pistón o percutor; este tipo tiene mayor demanda debido a su mayor eficiencia. A continuación se muestra la clasificación de este tipo de motores

Motor de engranaje

Tipo rotatorio

- Motor de veleta
- Motor de hélice
- Motor hidráulico
- Motor de leva excéntrica
- Pistón axial

Tipo oscilante

- Motor con eje inclinado
- Motor de engranaje

2.1.4.2.7. Motor con pistón eje inclinado

El aceite a presión que fluye desde la entrada empuja el pistón contra la brida y la fuerza resultante en la dirección radial hace que el eje y el bloque del cilindro giren en la dirección de la flecha. Este tipo de motor es muy conveniente para usos a alta presión y a alta velocidad. Es posible modificar su capacidad al cambiar el ángulo de inclinación del eje.

2.1.4.2.8. Motor oscilante con pistón axial

Tiene como función, el absorber un determinado volumen de fluido a presión y devolverlo al circuito en el momento que éste lo ejecute...

2.1.4.3. Actuadores neumáticos

A los mecanismos que convierten la energía del aire comprimido en trabajo mecánico se les denomina actuadores neumáticos. Aunque en esencia son idénticos a los actuadores hidráulicos, el rango de compresión es menor en este caso, además de que hay una pequeña diferencia en cuanto al uso y en lo que se refiere a la estructura, motivado a que los elementos de suministro de energía (aire) son diferentes de los empleados en los cilindros hidráulicos.

En esta clasificación aparecen los fuelles y diafragmas, que utilizan aire comprimido y son considerados como actuadores de simple efecto, y también los músculos artificiales de hule, que últimamente han recibido mucha atención.

De efecto simple

- Cilindro neumático
- Actuador neumático de efecto doble
- Actuador lineal de doble efecto sin vástago
- Con engranaje y cremallera
- Con engranaje y doble cremallera

- Motor neumático con veleta
- Con pistón
- Con una veleta a la vez
- Multiveleta
- Motor rotatorio con pistón
- De ranura vertical
- De émbolo
- Fuelles, diafragma y músculo artificial
- Cilindro de efecto simple

2.1.4.4. **Actuadores eléctricos**

La estructura de un actuador eléctrico es simple en comparación con la de los actuadores hidráulicos y neumáticos, ya que sólo requieren de energía eléctrica como fuente de energía. Como se utilizan cables eléctricos para transmitir electricidad y las señales, es altamente versátil y prácticamente no hay restricciones respecto a la distancia entre la fuente de energía y el actuador.

Existe una gran cantidad de modelos y es fácil utilizarlos con motores eléctricos estandarizados según la aplicación. En la mayoría de los casos es necesario utilizar reductores, debido a que los motores son de operación continua.

La forma más sencilla para el accionamiento con un pistón, seria la instalación de una palanca solidaria a una bisagra adherida a una superficie paralela al eje del pistón de accionamiento y a las entradas roscadas.

También existen los polímeros electroactivos, PEA (por su sigla en español) o EAP (por su sigla en inglés), los cuales son polímeros que usualmente cambian de forma o tamaño al ser estimulados por un campo eléctrico. Se

utilizan principalmente como actuadores, sensores, o la generación de músculos artificiales para ser empleados en robótica y en prostética.

2.1.5. **Relé**

El relé (en francés, relais, “relevo”) o relevador es un dispositivo electromagnético. Funciona como un interruptor controlado por un circuito eléctrico en el que, por medio de una bobina y un electroimán, se acciona un juego de uno o varios contactos que permiten abrir o cerrar otros circuitos eléctricos independientes. Fue inventado por Joseph Henry en 1835.

(https://es.wikipedia.org/wiki/Rel%C3%A9)

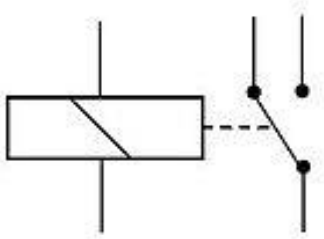


Figura 12: simbología de un rele

Fuente: es.wikipedia.org/wiki/Rel%C3%A

2.1.5.1. **Modulo rele de dos canales**

Cuando compremos un relé deberemos tener en cuenta el voltaje del aparato eléctrico que deseemos controlar ya que existen relés para 12 Voltios, 24 V, 110 V, 220 V, etc. Es importante que también tengas en cuenta de que lo que estás haciendo es peligroso, y por lo tanto hay que realizarlo con las medidas de seguridad adecuadas y nunca por niños que no estén acompañados por adultos.

2.1.5.2. Pines de conexión de relé

- COM- Pin común
- NC- Normalmente cerrado, en cuyo caso NC está conectado con COM cuando INT1 se establece bajo y se desconecta cuando INT1 es alto;
- NO- Normalmente abierto, en cuyo caso NO se desconecta con COM1 cuando INT1 se establece bajo y se conecta cuando INT1 es alto.
- El terminal 2 es similar al terminal 1, excepto que el puerto de control es INT2

- INT 1- Relé 1 puerto de control
- INT 2- Puerto de control del relé 2

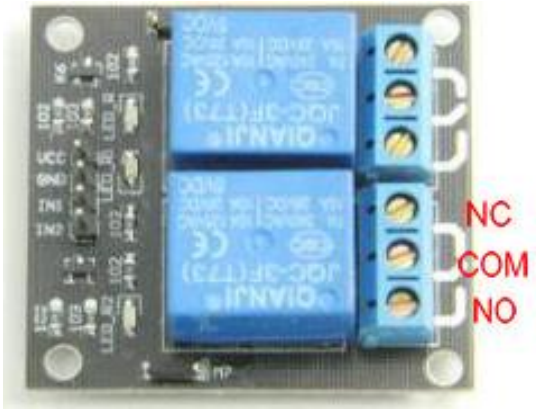


Figura 13: pines de conexión de un módulo relé

Fuente: www.geeetech.com/wiki/index.php/2-Channel_Relay_module

2.1.6. **Electroválvula**

Una electroválvula es una válvula electromecánica, diseñada para controlar el paso de un fluido por un conducto o tubería. La válvula se mueve mediante una bobina solenoide. Generalmente no tiene más que dos posiciones: abierto y cerrado, o todo y nada. Las electroválvulas se usan en multitud de aplicaciones para controlar el flujo de todo tipo de fluidos.

Una electroválvula tiene dos partes fundamentales: el solenoide y la válvula. El solenoide convierte energía eléctrica, mediante magnetismo, en energía mecánica para accionar la válvula.

Existen varios tipos de electroválvulas. En algunas electroválvulas el solenoide actúa directamente sobre la válvula dando la energía necesaria para su movimiento. También es posible construir electroválvulas biestables que usan un solenoide para abrir la válvula y otro para cerrar o bien un solo solenoide que abre con un impulso de corriente y cierra con el siguiente. Estas tienen dos contactos eléctricos, de modo que al cambiar de posición la válvula abre uno de ellos y cierra el otro.

2.1.6.1. Electroválvulas sencillas

Las electroválvulas de tipo directo pueden ser cerradas en reposo o normalmente cerradas lo cual quiere decir que cuando falla la alimentación eléctrica quedan cerradas o bien pueden ser del tipo abiertas en reposo o normalmente abiertas que quedan abiertas cuando no hay alimentación. Es decir, en el primer caso la válvula se mantiene cerrada por la acción de un muelle y el solenoide la abre venciendo la fuerza del muelle. Esto quiere decir que el solenoide debe estar activado y consumiendo energía mientras la válvula está abierta. Las normalmente abiertas, funcionan al revés.

Este tipo de válvulas se utilizan muy comúnmente en lavadoras, lavaplatos, riegos y otros usos similares.

2.1.6.2. Electroválvulas de tres vías

Hay electroválvulas que en lugar de abrir y cerrar lo que hacen es conmutar la entrada entre dos salidas, en una válvula de tres vías. Este tipo de electroválvulas a menudo se usan en los sistemas que tienen calefacción y preparación de agua caliente sanitaria lo que permite permutar el calentamiento de uno u otro sistema alternativamente utilizando una sola bomba de circulación.

En los calentadores de agua circulante, el agua se calienta según va pasando por el calentador en el momento del consumo y es la propia presión del agua la que abre la válvula del gas; pero en los calentadores por acumulación esto no es posible ya que el agua se calienta mientras está almacenada en un depósito y debe hacerlo aunque no haya circulación. Normalmente se utiliza una válvula solenoide, mandada por un termostato que, cuando detecta una temperatura por debajo de la de consigna (normalmente 60 ºC), desvía el agua caliente, destinada a la calefacción, por un intercambiador dispuesto en el depósito de agua caliente sanitaria y cuando el termostato determina que el agua ha llegado a la temperatura de acumulación, corta la corriente de la válvula, que vuelve a su posición de reposo, devolviendo el flujo de agua caliente al sistema de calefacción

Figura 14: electroválvula

Fuente:adajusa.es/electrovalvulas-de-control-de-procesos-y-domotica.com

III. MARCO METODOLÓGICO

3.1. Enfoque y diseño

- Enfoque mixto:
    - ✓ Cuantitativo
    - ✓ Cualitativo

3.2. Sujetos de la investigación

-
- Universo: Distrito de Piura - Perú.
- Población: Zonas residenciales, empresas e instituciones.

3.3. Métodos y procedimientos

Los pasos que se seguirán en el desarrollo de la investigación, en cumplimiento de los objetivos específicos, son los siguientes:

- Lectura e investigación de antecedentes , encontrándose trabajos relacionados con nuestro tema desarrollado
- Se investigo acerca de que animales son considerados domésticos
- Se recolecto información de horarios de alimentación y tipo de comida saludables, recomendada por expertos (veterinarias)
- Se recolecto de información para la viabilidad del módulo electrónico
- Se Realizó el diseño del prototipo del alimentador automático (estructura)
- Se programó los dispositivos (Arduino y sim900) para automatizar el alimentador de animales domésticos

- Evaluar el impacto saludable y económico producido por el alimentador automático

3.4. Técnicas e instrumentos

- Técnicas de muestreo: Simple. Se seleccionará un número determinado de zonas urbanas, rurales, instituciones de mayor interés de acuerdo con los Objetivos del Proyecto.
- Técnicas de recolección de datos: Encuestas, Internet, observación directa.
- Instrumentos de recolección de datos: Revisión de información estadística
- Confiabilidad y validez de los instrumentos: La información es proporcionada por entes públicos. Dicha información es confiable y válida, y por tanto no se requiere validación de algún especialista externo.

3.5. Aspectos éticos

Para el desarrollo de este trabajo de tesis se siguieron los aspectos éticos de acuerdo con el marco legal vigente de la oficina de Investigación de la Universidad Nacional de Piura y Ley del Procedimiento administrativo general y las normas legales de protección a los derechos del Autor indicadas para la realización original del mismo

4. Desarrollo del sistema

4.1. Descripción general

en la figura 15 se muestra el diagrama de bloques de prototipo de alimentador automático para animales domésticos utilizando una plataforma de módulo electrónico,, el cual se desarrollara con el módulo Arduino uno r3 como controlador de nuestro sistema , este módulo será el encargado de activar o desactivar los actuadores para el funcionamiento de nuestro alimentador automático

Así mismo la comunicación entre el alimentador y el usuario se realizara mediante mensaje de texto utilizando la tecnología GSM/GPRS que nos ofrece el módulo shield sim 900, permitiéndonos así activar o desactivar nuestro sistema , verificar el estado actual y programar la hora de alimentación de los animales domésticos

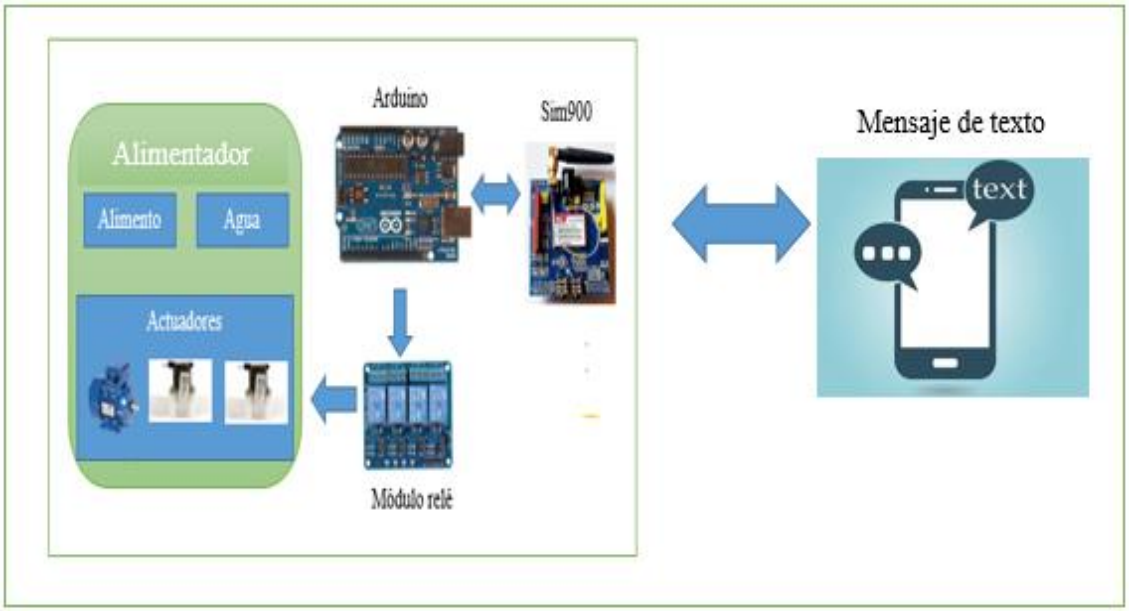


Figura 15: diagrama de bloques del prototipo de alimentador automático para animales domésticos utilizando una plataforma en módulo electrónico”

Fuente: propia

4.2. Componentes del sistema

4.2.1. Hardware

4.2.1.1. Arduino uno

El Arduino uno será el encargado de tomar decisiones, basadas en las ordenes automática que serán programadas y en ordenes enviadas por el usuario mediante mensaje de texto por medio del módulo de comunicación sim900

Utilizando los pines 3-4-5 para activar los modulos relay y los pines de comunicación 0 -1 en el cual ira conectado el modulo sim900

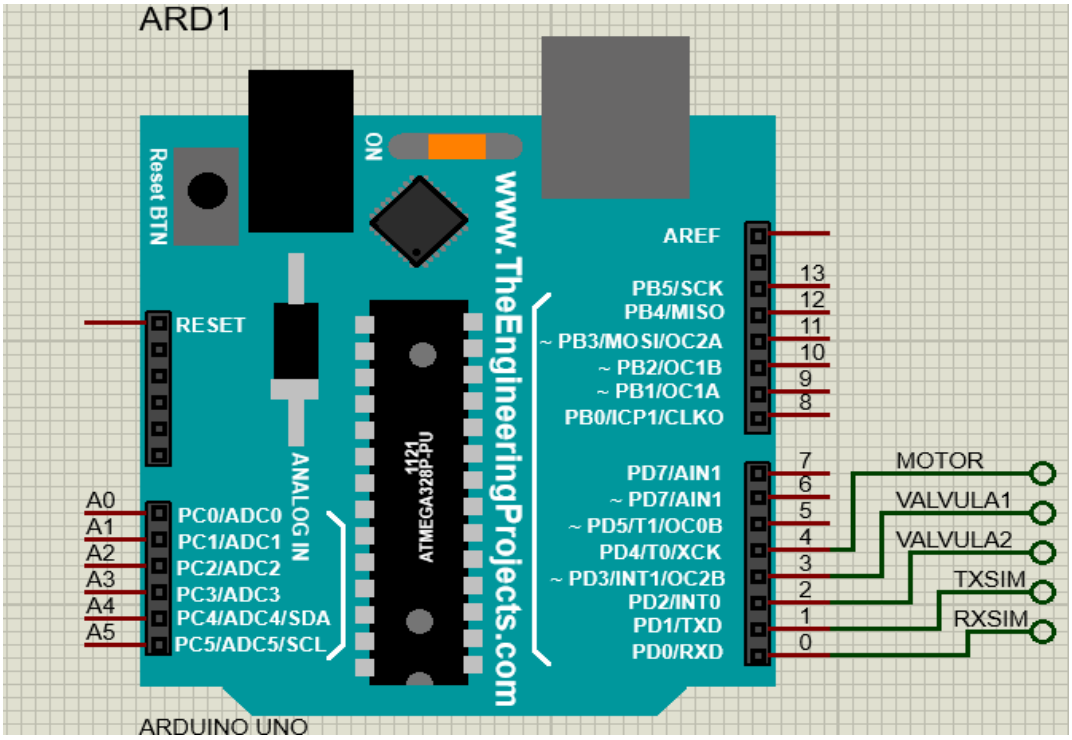
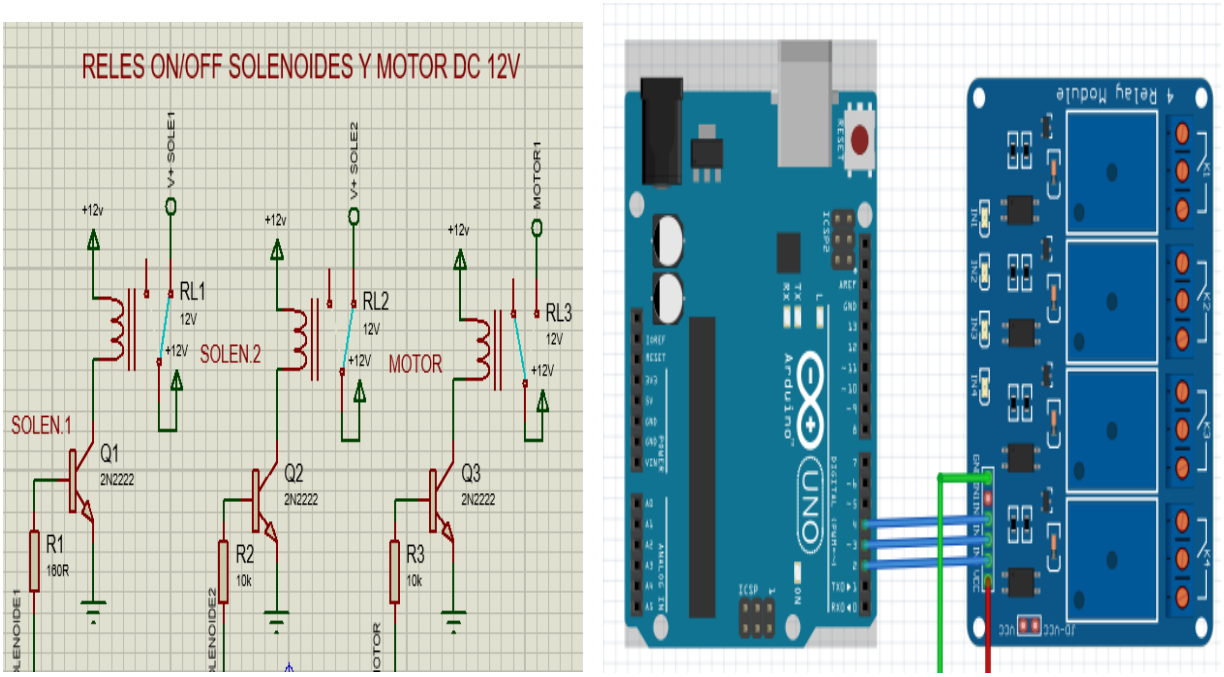


Figura 16: pines de conexión arduino

Fuente: propia

4.2.1.2. **Módulos relés**

Los módulos relés serán los encargados de la activar y desactivar el funcionamiento del motor y de las electroválvula se nuestro sistema, las cuales estarán programadas en el Arduino, para nuestro proyecto utilizaremos relay de 220 y 12 v



a) b)

Figura 17: a) simulación en proteus , b)diagrama de conexión Arduino modulo relé

Fuente: propia

4.2.1.3. **Módulo shield sim900**

El modulo SIM900 fundiona con tecnología gsm/gprs por el cual nos permitirá el envio y recepción de mensajes de texto ,este modulo esta encargado de enviar mediante mensaje el estado de nuestro sistema , asi como el de configurar el horario de alimentación de los animales domesticos , los pines de conexión en arduino será el 0 y el 1 tal como se muestra en la figura 18 de nuestra simulación

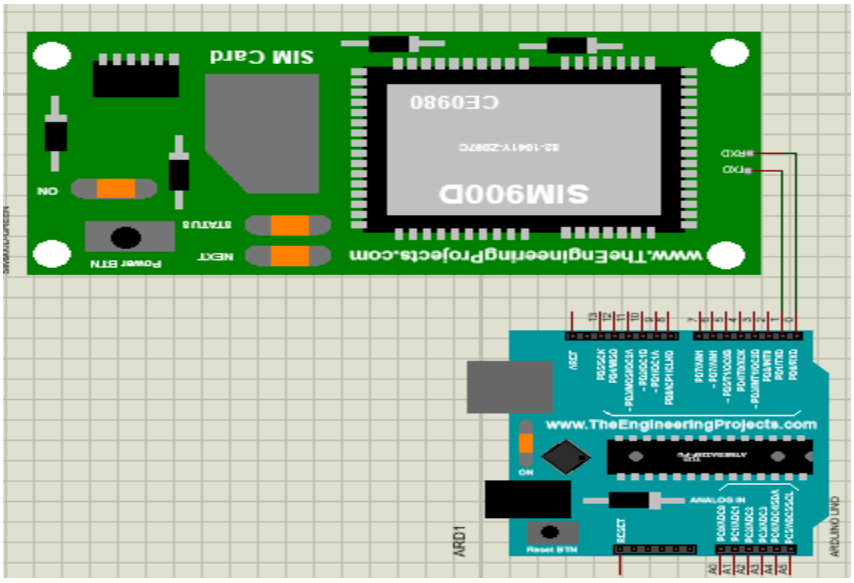


Figura 18: a) conexión en proteus entre SIM900 y Arduino uno

Fuente: propia

4.2.1.4. **Electroválvula**

En nuestro sistema la electroválvula toma un papel importante para el paso del agua de nuestro prototipo ala fuente del animal y ala vez para la eliminación del agua sobrante

La electroválvula a utilizar sera del Modelo **VALV-SOL-1D2-12V**

Figura 19: electroválvula 12v

Fuente:naylampmechatronics.com/

Especificaciones técnicas

- Voltaje de operación: 12V DC
- Corriente de operación: 0.6A
- Potencia consumo: 8W
- Temperatura de funcionamiento: 5ºC a 100ºC
- Presión de funcionamiento mínima: 0.02 MPa (0.2 Bar = 2.04 mca)
- Presión de funcionamiento máximo: 0.8 MPa (8 Bar = 81.6 mca)
- Tiempo de respuesta (apertura): ≤ 0.15 s
- Tiempo de respuesta (cerrado): ≤ 0.3 s
- Conector tuberia: Rosca externa 1/2" NPS Macho
- Reposo: Normalmente cerrado
- Tipo de válvula: Diafragma
- Adecuado para agua y fluidos de baja viscosidad
- Dimensiones: 60*85*26mm
- No se recomienda para aplicaciones que usan solo la gravedad, por la presión mínima de funcionamiento

4.2.2. **Software**

4.2.2.1. **Entorno de Programación de Arduino (IDE)**

IDE – entorno de desarrollo integrado, llamado IDE (sigla en inglés de integrated development environment), es un programa informático compuesto por un conjunto de herramientas de programación. Puede dedicarse en exclusiva a un solo lenguaje de programación o bien puede utilizarse para varios.

Un IDE es un entorno de programación que ha sido empaquetado como un programa de aplicación, es decir, que consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica (GUI). Además en el caso de Arduino incorpora las herramientas para cargar el programa ya compilado en la memoria flash del hardware a través del puerto serie.

Los programas de arduino están compuestos por un solo fichero con extensión “ino”, aunque es posible organizarlo en varios ficheros. El fichero principal siempre debe estar en una carpeta con el mismo nombre que el fichero.

Anteriormente a la versión 1.x de Arduino se usaba la extensión “pde”. Cuando se pasó a la versión 1.x hubo grandes cambios, que deben tenerse en cuenta si se usa código antiguo.

La última versión del IDE de Arduino es la 1.6.8. Los grandes cambio del IDE Arduino se produjo en el cambio de la versión 0.22 a la 1.0 y posteriormente en el cambio de la versión 1.0.6 a la 1.6.0 con grandes mejoras en el IDE de Arduino.

El el caso de la versión 1.6.0 los cambios han sido principalmente internos más que en el aspecto de la herramienta. También es destacable desde la aparición de la versión 1.6.2 la incorporación de la **gestión de librerías** y la **gestión de placas,** muy mejoradas respecto a las versiones anteriores y avisos de actualización de versiones de librerías y cores.

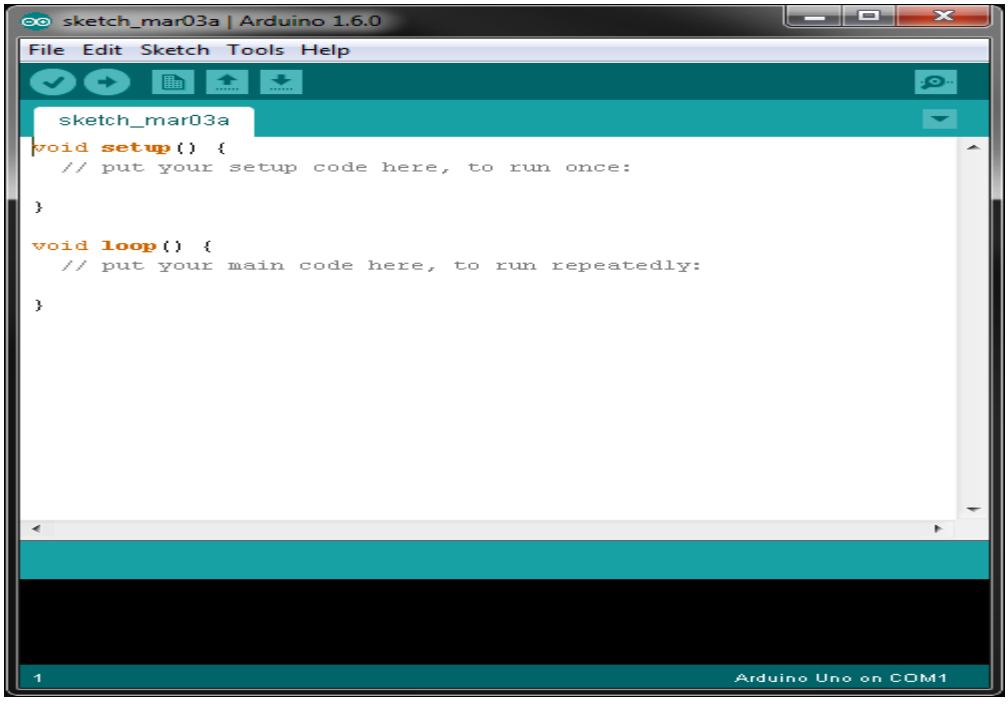


Figura 20: interface ide arduino

Fuente:aprendiendoarduino.wordpress.com/

4.2.2.2. **Software proteus**

Se trata de un completo programa que permite diseñar y simular circuitos electrónicos de forma práctica y accesible.

A todos aquellos que trabajen en el ámbito de la electrónica les interesará la aplicación Proteus. Se trata de un completo programa que permite diseñar y simular circuitos electrónicos de forma práctica y accesible. Proteus está formado por dos utilidades principales: ARES e ISIS, y por los módulos Electra y VSM. Si necesitas crear componentes con Proteus e ISIS será una tarea fácil. Prueba las herramientas ARES e ISIS de Proteus al descargar el programa.

Principales características

- La aplicación ISIS permite generar circuitos reales, y comprobar su funcionamiento en un PCB (*printed circuit board*).
- Entorno de diseño gráfico de esquemas electrónicos fácil de utilizar y con efectivas herramientas.
- Entorno de simulación con la tecnología exclusiva de Proteus de modelación de sistemas virtuales (VSM).
- Herramienta ARES para el enrutado, ubicación y edición de componentes, utilizado para la fabricación de placas de circuito impreso.
- Interfaz intuitivo y atractivo estandarizado para todos los componentes de Proteus.

Proteus cuenta con una gran cantidad de funciones para trabajar con circuitos electrónicos. Por ejemplo, permite generar pistas de cobre de forma automática. Además, permite la simulación de PICs casi a tiempo real, de forma que podemos comprobar si el circuito creado funciona de la forma que esperábamos.

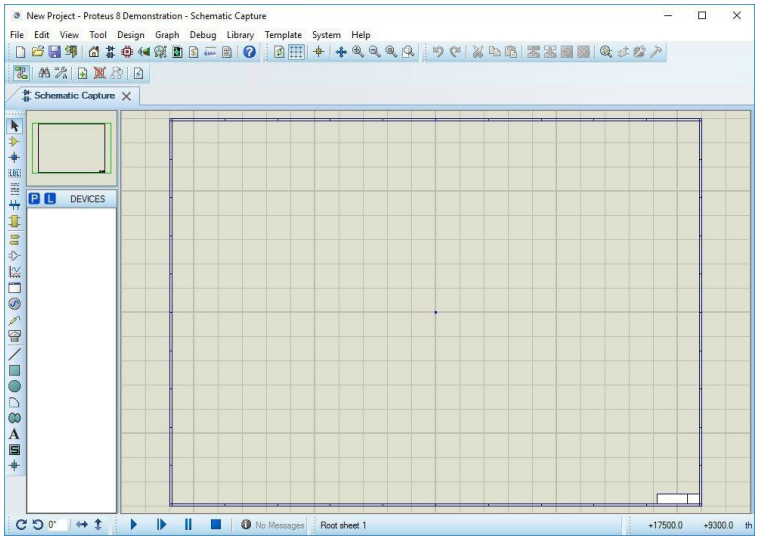


Figura 21: interface software proteus

Fuente propia

4.2.2.3. **Hyperteminal**

HyperTerminal es un programa que se puede utilizar para conectar con otros equipos, sitios Telnet, sistemas de boletines electrónicos (BBS), servicios en línea y equipos host, mediante un módem, un cable de módem nulo o Ethernet.

Aunque utilizar HyperTerminal con un servicio de boletín electrónico para tener acceso a información de equipos remotos es una práctica que está dejando de ser habitual gracias al World Wide Web, HyperTerminal sigue siendo un medio útil para configurar y probar el módem o examinar la conexión con otros sitios. Para obtener más información, consulte Configurar una conexión nueva.

HyperTerminal graba los mensajes enviados o recibidos por servicios o equipos situados al otro extremo de la conexión. Por esta razón, puede actuar como una valiosa herramienta para solucionar problemas de configuración y uso del módem. Para confirmar que el módem está bien conectado o ver su configuración, puede enviar comandos a través de HyperTerminal y ver los resultados. HyperTerminal ofrece la funcionalidad de desplazamiento, que le permite revisar el texto recibido que sobrepase el espacio de la pantalla.

HyperTerminal sirve también para transferir archivos grandes de un equipo a un equipo portátil a través del puerto serie, en lugar de realizar la configuración del portátil en una red. Para obtener más información, consulte Enviar un archivo a un equipo remoto.

Puede utilizar HyperTerminal para ayudar a depurar el código fuente desde un terminal remoto. También puede utilizar HyperTerminal para comunicarse con los equipos antiguos basados en caracteres.

HyperTerminal está diseñado para ser una herramienta fácil de utilizar y no viene a sustituir a otras herramientas principales disponibles en el mercado. HyperTerminal puede utilizarse para realizar las tareas específicas descritas, pero no intente utilizarlo para necesidades de comunicación más complejas. Para obtener más información acerca de las funciones de HyperTerminal, consulte la lista de preguntas más frecuentes de HyperTerminal en el sitio Web de Hilgraeve.(http://redesparacomputadores-lubian.blogspot.com/)

Figura 22: interface software hyperterminal

Fuente propia

4.3. Etapa de diseño

4.3.1. Estructura del alimentador automático

Para la elaboración el sistema de alimentador automático para animales domésticos, se realizó en estructura metálica, el cual se procedió a elaborarlo en tres etapas , la primera etapa se procedido a elaborar la base de nuestro prototipo como la colocación del recipiente que contendrá la comida y el recipiente de agua como se puede observar en la figura 23

Figura 23: primera etapa de la estructura del alimentador automático

(Contenedor de alimento y agua)

Fuente: propia

En la segunda etapa de nuestro prototipo se realizó el acondicionamiento para el transporte de la comida, utilizado así perno helicoidal para que en giro transporte la comida por la tubería hasta llega al recipiente final como se muestra en la figura 24:

Figura 24: segunda etapa de la estructura del alimentador automático

(Tubería de transporte)

Fuente: propia

En la tercera se procedió a realizar a darle los acabados finales, cubriendo nuestro sistema de material (latón), y terminándolo con su respectivo pintado tal como se muestra en la figura25

Figura 25: tercer etapa de la estructura del alimentador automático

(Acabados finales)

Fuente: propia

4.3.2. **Configuración del módulo SIM900**

Para poder comunicarnos vía comandos AT tendremos que cargar un programa para la comunicación serie para poder utilizarlo con el arduino en el cual . Crearemos una instancia llamada SIM900 y seleccionaremos los pines del Arduino que queramos usar para comunicarnos *(Rx y Tx)*. Nosotros hemos elegido el 7 y el 8, pero podéis usar cualquiera que sea compatible con la librería. También podéis cambiar la velocidad de comunicación, pero debe ser la misma para el puerto serie y para la instancia que hemos creado. Nosotros hemos elegido 19200 porque es la que usa el SIM900, de forma que podamos usar los programas que ya tenemos de esas sesiones.

```
#include <SoftwareSerial.h>
SoftwareSerial SIM900(0, 1);
void setup(){
 SIM900.begin(19200);
 Serial.begin(19200);
 delay(1000);
}
void loop(){
 //Envíamos y recibimos datos
 if (Serial.available() > 0)
 SIM900.write(Serial.read());
 if (SIM900.available() > 0)
 Serial.write(SIM900.read());
}
```

Figura 26: código para pruebas del módulo sim900

Fuente: propia

Una vez hayamos cargado el programa abrimos el monitor serie y seleccionamos la velocidad correcta. El primer comando AT nos servirá simplemente para saber si el módulo responde y que por lo tanto la comunicación funciona. Y este comando es simplemente AT, lo escribimos y pulsamos *INTRO*. Debería respondernos con un OK; si no deberíamos repasar que esté todo en orden: conexiones, encendido y velocidad correcta.

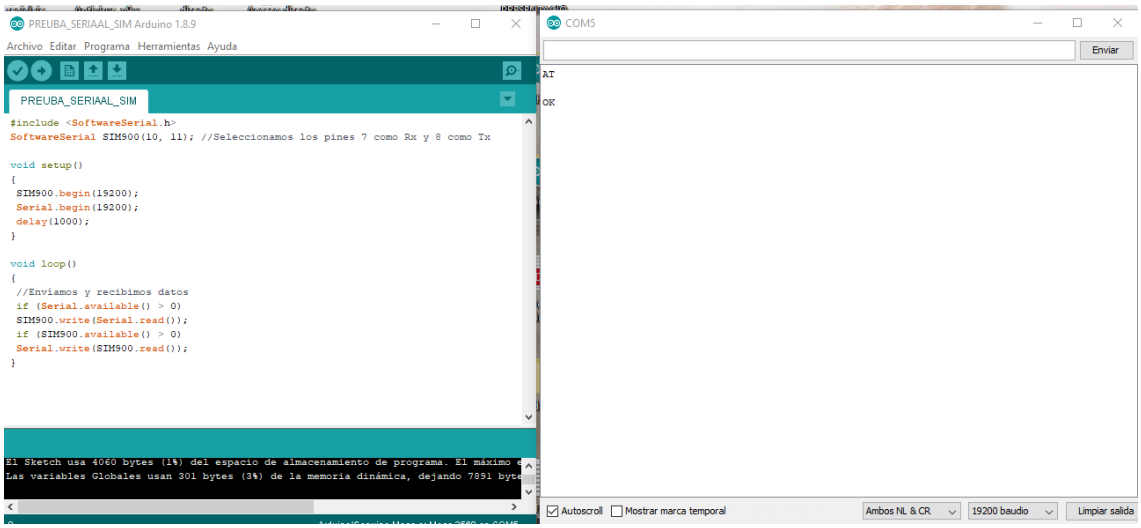


Figura 27: prueba de los comandos AT

Fuente: propia

Procedemos a ingresar los principales comando at para verificar el correcto funcionamiento de nuestro sim900

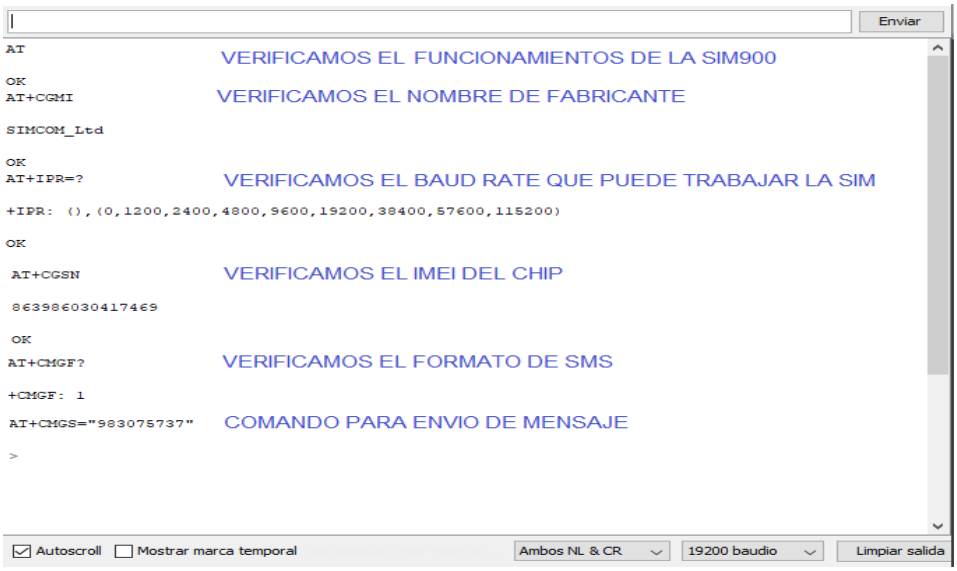


Figura 28: respuestas de comandos at

Fuente: propia

4.3.3. **Simulación de módulo relay Arduino**

```
int relay1 = 2;
int relay2 = 3;
int relay3= 4;
void setup() {
Serial.begin(9600);
  pinMode(relay1, OUTPUT);
  pinMode(relay2, OUTPUT);
  pinMode(relay3, OUTPUT);
}
void loop() {
  digitalWrite(relay1, HIGH);   // Envia el valor HIGH (5V) al pin digital relay
  Serial.println("rele1 activo");
  delay(2000);
  digitalWrite(relay1, LOW);    // Envia el valor LOW (0V) al pin digital relay
  delay(10000);
   Serial.println("rele1 desactivado");// Espera 10 segundos
digitalWrite(relay2, HIGH);   // Envia el valor HIGH (5V) al pin digital relay
   Serial.println("rele2 activo");
  delay(2000);                // Espera 2 segundos
  digitalWrite(relay2, LOW);    // Envia el valor LOW (0V) al pin digital relay
  delay(10000);
  Serial.println("rele2 desactivado");
digitalWrite(relay3, HIGH);   // Envia el valor HIGH (5V) al pin digital relay
   Serial.println("rele3 activo");
  delay(2000);                // Espera 2 segundos
  digitalWrite(relay3, LOW);    // Envia el valor LOW (0V) al pin digital relay
  delay(10000);
Serial.println("rele3 desactivado");
}
```

Figura 29: Código de programación para funcionamiento los módulos relés

Fuente:propia

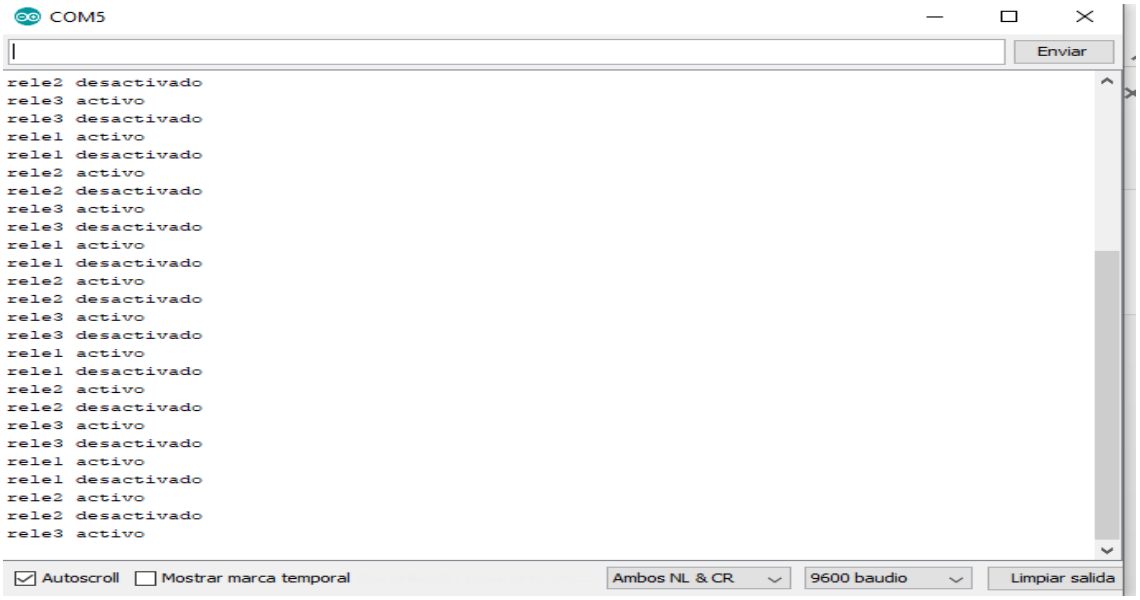


Figura 30: Visualización de datos mediante puerto serial

Fuente:propia

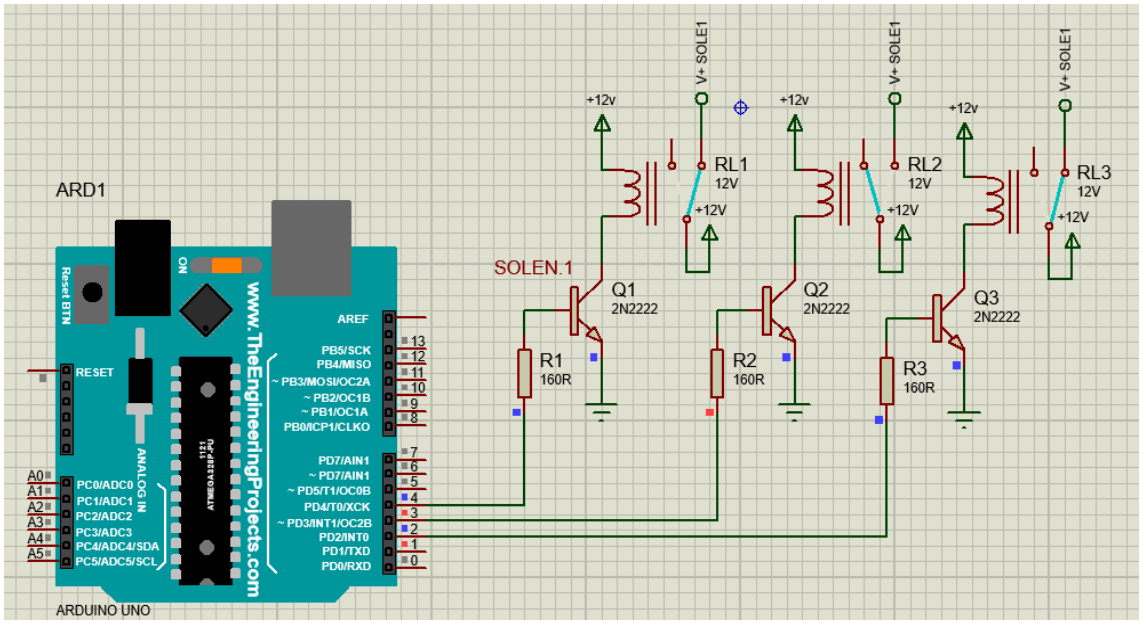


Figura 31: simulación en proteus

Fuente: propia

4.4. Pruebas y resultados

4.4.1. **FOTOS DEL PROYECTO**

Figura 32: prototipo de alimentador automático para animales domésticos utilizando una plataforma en modulo electrónico

Fuente: propia

Figura 33: interior del prototipo de alimentador automático para animales domésticos

Fuente: propia

Figura 34: panel de control del prototipo de alimentador automático para animales domésticos

Fuente: propia

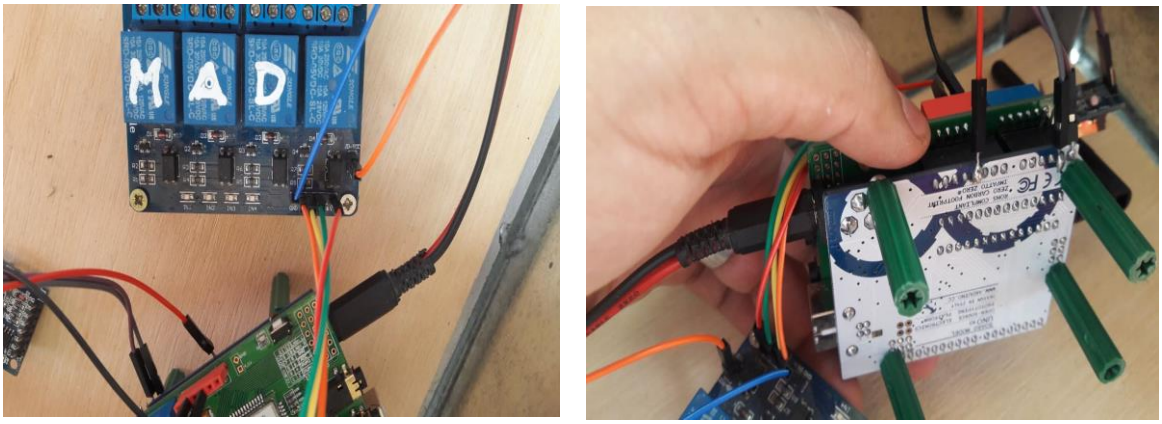


Figura 35: módulos de nuestro panel de control del prototipo de alimentador automático para animales domésticos

Fuente propia

4.4.2. **Diseño realizado en software proteus**

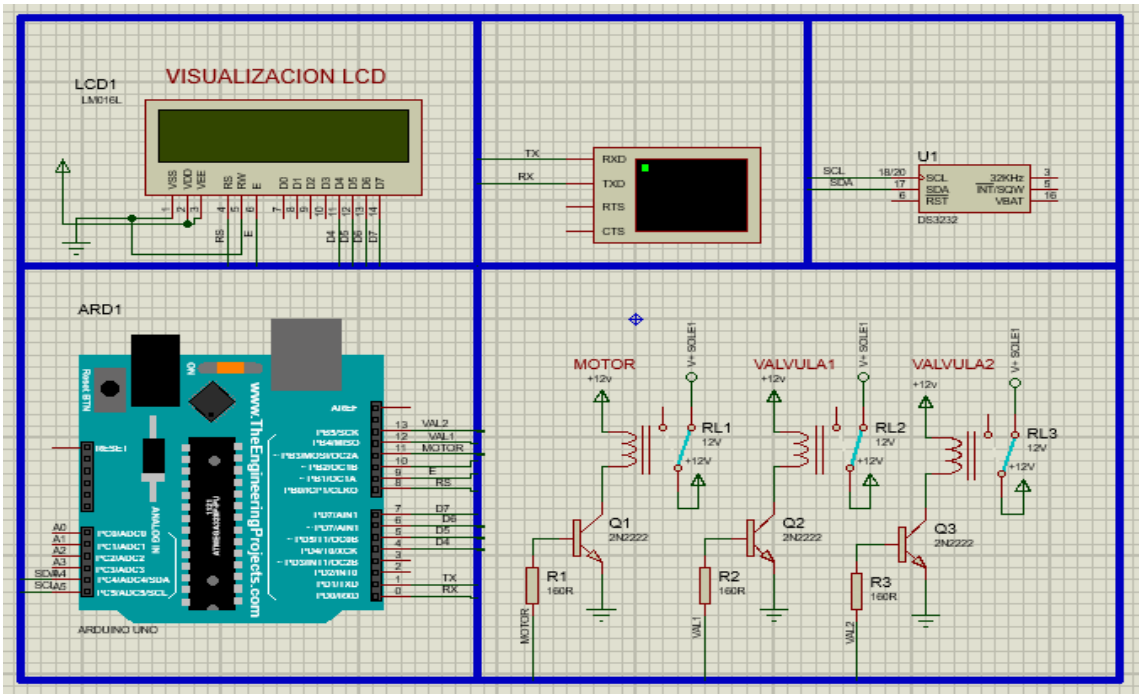


Figura 36:diseño de nuestro sistemade alimentador automatico

Fuente propia

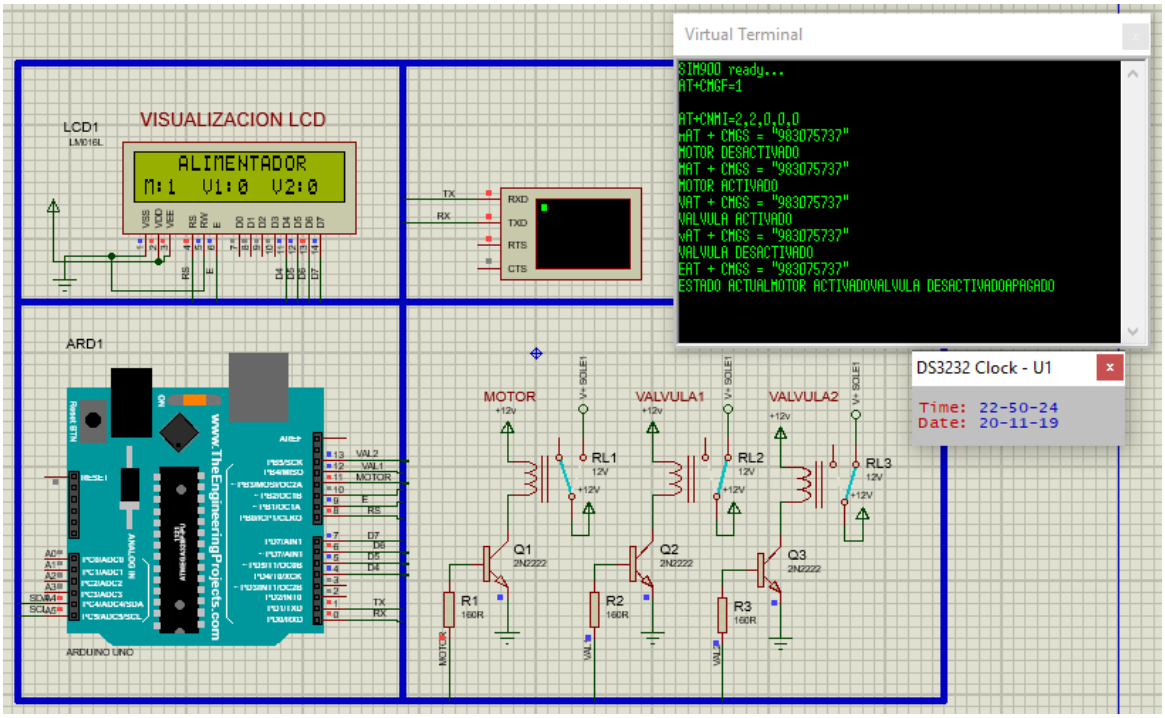
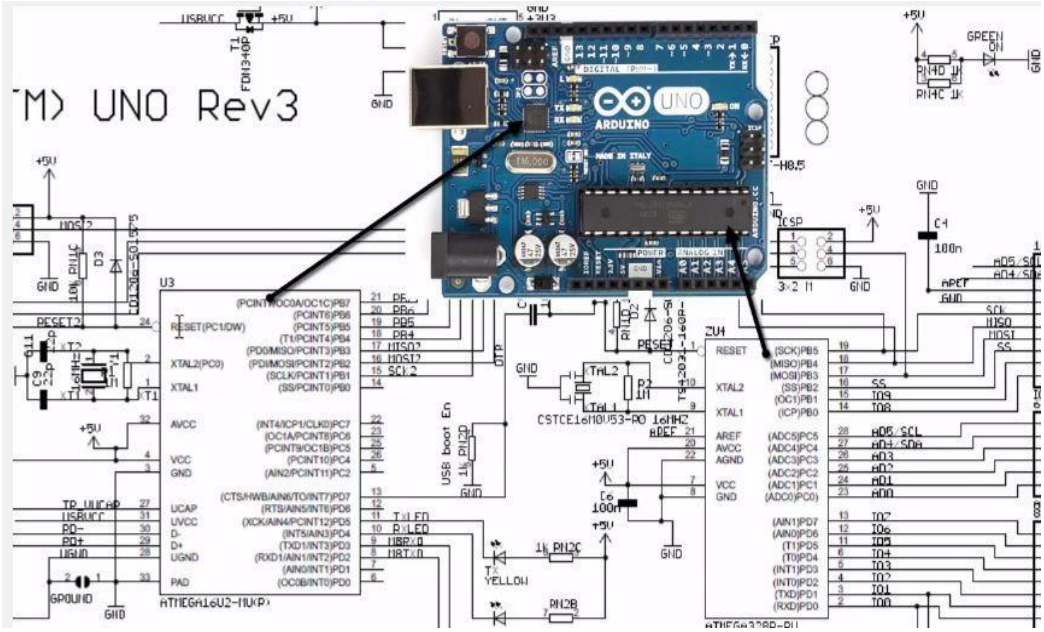


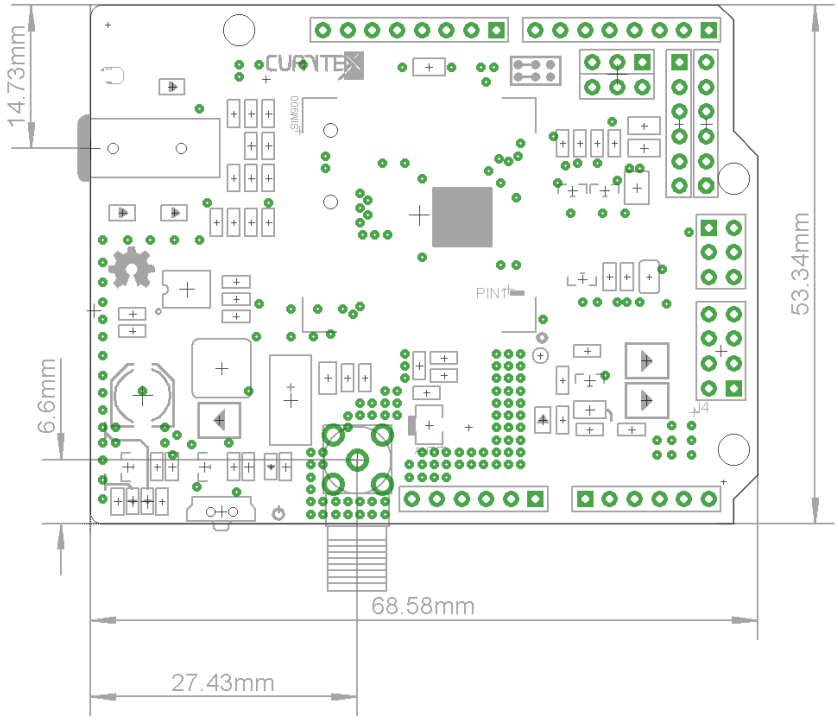
Figura 37: simulación de nuestro sistema de alimentador automático

Fuente propia

4.4.3. **Esquemtico de tarjeta arduino**



4.4.4. **Esquemático de tarjeta sim900**



4.4.5. **Código de programación del prototipo de alimentador automático para animales domésticos**

```
#include <LiquidCrystal.h>
#include "Sodaq_DS3231.h"
#include <Wire.h>
LiquidCrystal lcd(8, 9, 4, 5, 6, 7);
int motor = 11;
int agua = 12;
int desague= 13;

char DiaSemana[][4] = {"Dom", "Lun", "Mar", "Mie", "Jue", "Vie", "Sab"};
String numero_destino = "983075737";  // número registrado por software.
String estado  ="APAGADO\n";
String estado1 ="APAGADO\n";
String estado2 ="APAGADO\n";
String L1="0";
String L2="0";
String L3="0";
String textMessage;

void setup(){

 lcd.begin(16, 2);
 Serial.begin(19200);
 rtc.begin();
 Wire.begin();
```

```
  Serial.println("SIM900 ready...");
  delay(2000);

  Serial.println("AT+CMGF=1\r");
  delay(100);
  Serial.print("AT+CNMI=2,2,0,0,0\r");
  delay(100);

  pinMode(motor, OUTPUT);
  pinMode(agua, OUTPUT);
  pinMode(desague, OUTPUT);
  digitalWrite(motor, LOW);
  digitalWrite(agua, LOW);
  digitalWrite(desague, LOW);
  }
void loop()
{

horario();

   lcd.setCursor(3,0);
   lcd.print("ALIMENTADOR");
   lcd.setCursor(0,1);
   lcd.print("M:");
   lcd.print(L1);
   lcd.setCursor(5,1);
   lcd.print("V1:");
   lcd.print(L2);
   lcd.setCursor(11,1);
   lcd.print("V2:");
```

```
    lcd.print(L3);
 if(Serial.available()>0){
   textMessage = Serial.readString();
   Serial.print(textMessage);
   delay(10);
  }
  if(textMessage.indexOf("M")>=0){
   digitalWrite(motor, HIGH);
   L1="1";
   estado = "MOTOR ACTIVADO\n";
   Send_message(estado);  // Función de transmisión de datos.
   textMessage = "";
   delay(500);
   return;
  }
if(textMessage.indexOf("m")>=0){
  digitalWrite(motor, LOW);
  L1="0";
  estado = "MOTOR DESACTIVADO\n";
  Send_message(estado);  // Función de transmisión de datos.
  textMessage = "";
   delay(500);
  return;
   }
 if(textMessage.indexOf("V")>=0){
   digitalWrite(agua, HIGH);
   L2="1";
   estado1 = "VALVULA ACTIVADO\n";
   Send_message(estado1);  // Función de transmisión de datos.
   textMessage = "";
```

```
   delay(500);
    return;
  }
 if(textMessage.indexOf("v")>=0){
    digitalWrite(agua, LOW);
    L2="0";
    estado1 = "VALVULA DESACTIVADO\n";
    Send_message(estado1);  // Función de transmisión de datos.
   textMessage = "";
   delay(500);
   return;
  }
 if(textMessage.indexOf("U")>=0){
    digitalWrite(desague, LOW);
    L3="0";
    estado2 = "VALVULA 1 DESACTIVADO\n";
    Send_message(estado2);  // Función de transmisión de datos.
    textMessage = "";
    delay(500);
    return;
  }
if(textMessage.indexOf("u")>=0){
    digitalWrite(desague, HIGH);
    L3="1";
    estado2 = "VALVULA1 ACTIVADO\n";
    Send_message(estado2);  // Función de transmisión de datos.
    textMessage = "";
    delay(500);
    return;
    }
```

```
  if(textMessage.indexOf("E")>=0){
    String prueba= (estado  + estado1 + estado2);
   String message="ESTADO ACTUAL\n"+prueba;
   Send_message(message);
    textMessage = "";
    delay(500);
    return;
  }
 if(textMessage.indexOf("#ACTIVAR")>=0){
   comida();
   String message="ALIMENTACION EN PROCES0";
   comida();
   Send_message(message);
   textMessage = "";
    delay(500);
    return;
  }
  }
void Send_message(String estado){

  Serial.println("AT + CMGS = \"" + numero_destino +"\"");
  delay(100);
  Serial.print(estado);
  delay(100);
  //Serial.println(estado);
  Serial.print((char)26);
  delay(100);
  //desplazamiento_lcd();
}
```

```
void comida(){
  digitalWrite(motor, HIGH);   // ACTIVA MOTOR
  delay(1500);              // wait for a second
  Serial.println(" ACTIVA MOTOR");
  digitalWrite(motor, LOW);    // DESACTIVA MOTOR
  delay(1000);
  Serial.println("DESACTIVA MOTOR.");
  digitalWrite(desague, HIGH);   // VALVULA DESFOGE ON
  digitalWrite(agua, LOW); //VALVULA ENTRADA DE AGUA OFF
  delay(1000);              // wait for a second
  Serial.println("VALVULA DESFOGE ON");
  Serial.println("VALVULA ENTRADA DE AGUA OFF");
  digitalWrite(desague, LOW);    // VALVULA DESFOGE OFF
  digitalWrite(agua, HIGH); //VALVULA ENTRADA DE AGUA ON
  delay(2000);
  digitalWrite(agua, LOW); //VALVULA ENTRADA DE AGUA OFF
  delay(1000);
  return;}
void horario() {
  DateTime now = rtc.now();
  int hora    =now.hour();
  int minuto  =now.minute();
   int segundo = now.second();
  if(hora ==21 &&  minuto ==02 && segundo ==15 ){
  comida();
  return;
  //delay(5000);
   }}
```

Figura 38: código de programación del prototipo de alimentador automático para animales domésticos

Fuente propia

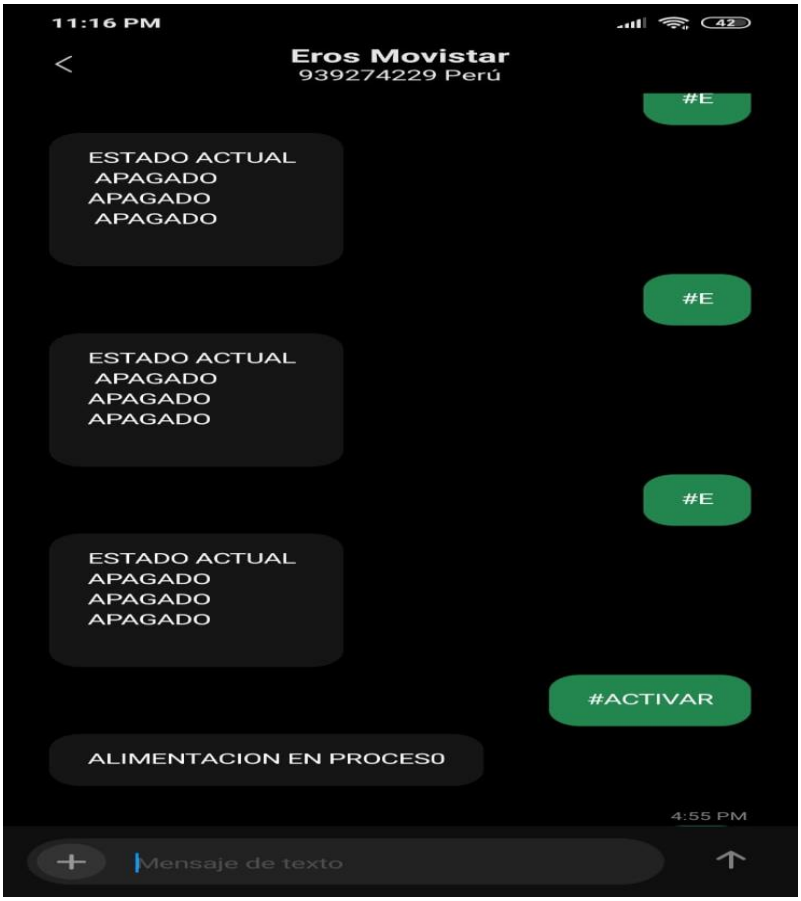


figura 39: envio y recepción de mensajes de texto del estado se nuestro sistemas

Fuente propia

CONCLUSIONES

- Este sistema se diseñó com la finalidad con la finalidad de que estos animales sean alimentados correctamente , debidos a que muchas veces la falta de tiempo de sus dueños , por diversos motivos , hacen que estos no coman a su horario correspondiente , perjudicándolos asi en su nutrición, nuestro proyecto de tesis ayudara a solucionar este problema , pues nos permitirá tener un control en el horario y en las cantidades de alimento correspondiente de estos animales domésticos.

- El sistema quedó compuesto por un módulo electrónico Arduino encargado de controlar y activar los dispositivos encargados de que la comidad llegue a la fuente final

- El sistema de comunicaciones quedo compuesto por el módulo shiel sim900, utilizando la tecnología gsm /gprs , mediante el cual mantendremos un monitoreo del estado de nuestro sistema automático y a la vez permitiéndonos realizar acciones en nuestro sistema mediante mensajes de texto

- ✓ El desarrollo de nuestro sistema se desarrolló con ayuda de los software proteus y el software ide de Arduino, permitiéndonos simular y programar nuestro sistema automático de alimentación para animales domésticos.

RECOMENDACIONES

- Para el correcto funcionamiento de nuestro sistema de comunicaciones, se debe tener en cuenta la ubicación de nuestro dispositivo y verificar que cuente con saldo disponible para el envío de mensajes
- se recomienda enfocarse más en el módulo sim900 , ya que este proyecto solamente se utilizado para el envío y recepción de mensajes, , este módulo es muy interesante ya que nos permite también realizar llamadas y conectarse a la red

- Se recomienda para el diseño de control del horario de alimentación, tener encuentra la edad del animal doméstico que se desea alimentar con este proyecto
- Se recomienda definir estratégicamente la ubicación del prototipo a desarrolla en un futuro, teniendo en cuenta su diseño de la estructura final del alimentador automático de animales domésticos
- Se recomienda tener mucho cuidado con la compra de los alimentos para los animales domésticos, debido a que cada animal tiene un tipo de alimento diseñado exclusivamente para su especie
- se recomienda tener mucho higiene en la manipulación de los alimentos al momento de la colocación en el depósito del prototipo

REFERENCIAS BIBLIOGRAFICAS

Guido Antonio Acosta Laso, (2012). “Proyecto De Factibilidad Para La Creación De Un Dispensador De Alimentos Automatizado Para Perros En El Distrito Metropolitano De Quito” “*PONTIFICIA UNIVERSIDAD CATÓLICA DEL ECUADOR* obtenido en : https://docplayer.es/12160854-Pontificia-universidad-catolica-del-ecuador-facultad-de-ciencias-administrativas-y-contables.html

Franco guzmán, Luis Fernando, Galicia Jiménez, Jesús Rodolfo y ostria valle diana. (2010). “Desarrollo De Un Sistema De Dosificación Automático De Alimentos Para Equinos” *INSTITUTO POLITECNICO NACIONAL*Obtenido de: https://tesis.ipn.mx/handle/123456789/9820

Ruben adrian de la mara(2017).”Arduino+ módulo gsm/gprspara moniotizacion y gestión remota en un viñedo.”*UNIVERSITAT OBERTA DE CATALUNYA* http://openaccess.uoc.edu/webapps/o2/bitstream/10609/65345/6/radriandTFG0617memoria.pdf

María Estela Raffino(2019) de argentina ,” Animales Domésticos"” obtenido de : https://concepto.de/animales-domesticos/

Alimentación de los animales domésticos (SF) obtenido de : https://www.queanimal.com/que-comen-los-animales-domesticos/

shangai SIMcom wireless solutions ltd.(2010) obtenido

http://biblioteca.geekfactory.mx/Modulo%20Simcom%20SIM900%20GSM/DOC_SIM900_AT_Command_Manua.pdf

Mar Domínguez & Co.(2014) "modulo rele dc" obtenido de :

https://soloarduino.blogspot.com/2014/01/modulo-keyes-rele-sr1y.html

jean francois pillou (2008). "Estándar GPRS (Servicio general de paquetes de radio)" obtenido en:

https://es.ccm.net/contents/680-estandar-gprs-servicio-general-de-paquetes-de-radio

carlos villagomez(2017)." Estándar GSM (Sistema global de comunicaciones móviles).obtenido en:

https://es.ccm.net/contents/681-estandar-gsm-sistema-global-de-comunicaciones-moviles

hetpro." SIM900 GSM GPRS SHIELD CON ARDUINO UNO" obtenido en :

https://www.instructables.com/id/SIM900-GSM-GPRS-SHIELD-CON-ARDUINO-UNO/

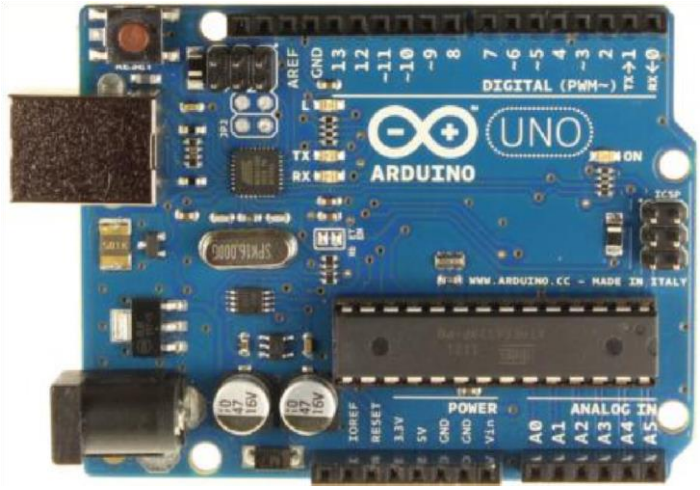
Ivan Uriarte" MÓDULO GSM/GPRS: LLAMAR Y ENVIAR SMS" obtenido en https://www.prometec.net/gprs-llamar-enviar-sms/

electronicaestudio(sf) obtenido en :
http://www.electronicaestudio.com/docs/ISTD-034.pdf

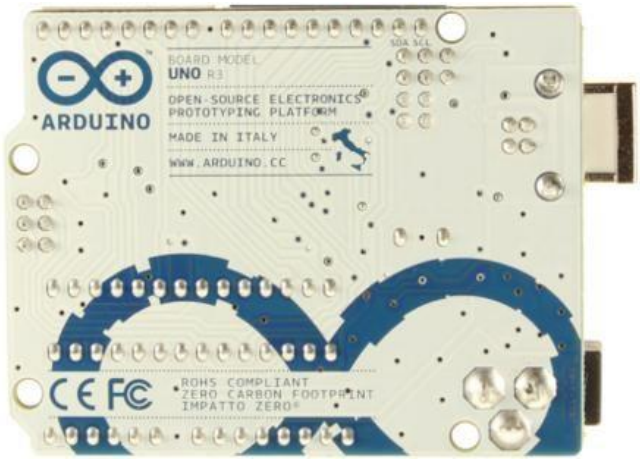
seeed technology co(2008)." GPRS Shield V2.0 obtenido en :
http://wiki.seeedstudio.com/GPRS_Shield_V2.0/

ANEXOS

Arduino Uno



Arduino Uno R3 Front

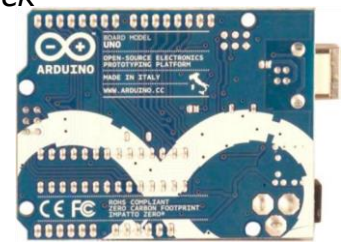


Arduino Uno R3 Back

Arduino Uno R2 Front

Arduino Uno SMD

Arduino Uno Front



Arduino Uno Back

5. Overview

The Arduino Uno is a microcontroller board based on the ATmega328 (datasheet). It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz ceramic resonator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started.

The Uno differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it features the Atmega16U2 (Atmega8U2 up to version R2) programmed as a USB-to-serial converter.

Revision 2 of the Uno board has a resistor pulling the 8U2 HWB line to ground, making it easier to put into DFU mode.

Revision 3 of the board has the following new features:

- 1.0 pinout: added SDA and SCL pins that are near to the AREF pin and two other new pins placed near to the RESET pin, the IOREF that allow the shields to adapt to the voltage provided from the board. In future, shields will be compatible both with the board that use the AVR, which operate with 5V and with the Arduino Due that operate with 3.3V. The second one is a not connected pin, that is reserved for future purposes.
- Stronger RESET circuit.
- Atmega 16U2 replace the 8U2.

"Uno" means one in Italian and is named to mark the upcoming release of Arduino 1.0. The Uno and version 1.0 will be the reference versions of Arduino, moving forward. The Uno is the latest in a series of USB Arduino boards, and the reference model for the Arduino platform; for a comparison with previous versions, see the index of Arduino boards.

6. Summary

Microcontroller	ATmega328
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
Analog Input Pins	6
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328) of which 0.5 KB used by bootloader
SRAM	2 KB (ATmega328)
EEPROM	1 KB (ATmega328)
Clock Speed	16 MHz

7. Schematic & Reference Design

EAGLE files: arduino-uno-Rev3-reference-design.zip (NOTE: works with Eagle 6.0 and newer) Schematic: arduino-uno-Rev3-schematic.pdf

Note: The Arduino reference design can use an Atmega8, 168, or 328, Current models use an

ATmega328, but an Atmega8 is shown in the schematic for reference. The pin configuration is identical on all three processors.

8. Power

The Arduino Uno can be powered via the USB connection or with an external power supply. The power source is selected automatically.

External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm center-positive plug into the board's power jack. Leads from a battery can be inserted in the Gnd and Vin pin headers of the POWER connector.

The board can operate on an external supply of 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may be unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts. The power pins are as follows:

- **VIN.** The input voltage to the Arduino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.
- **5V.**This pin outputs a regulated 5V from the regulator on the board. The board can be supplied with power either from the DC power jack (7 - 12V), the USB connector (5V), or the VIN pin of the board (7-12V). Supplying voltage via the 5V or 3.3V pins bypasses the regulator, and can damage your board. We don't advise it.
- **3V3.** A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.
- **GND.** Ground pins.

9. Memory

The ATmega328 has 32 KB (with 0.5 KB used for the bootloader). It also has 2 KB of SRAM and 1 KB of EEPROM (which can be read and written with the EEPROM library).

10. Input and Output

Each of the 14 digital pins on the Uno can be used as an input or output, using pinMode(), digitalWrite(), and digitalRead() functions. They operate at 5 volts. Each pin can provide or receive a maximum of 40 mA and has an internal pull-up resistor (disconnected by default) of 20-50 kOhms. In addition, some pins have specialized functions:

- **Serial: 0 (RX) and 1 (TX).** Used to receive (RX) and transmit (TX) TTL serial data. These pins are connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip.
- **External Interrupts: 2 and 3.** These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the attachInterrupt() function for details.
- **PWM: 3, 5, 6, 9, 10, and 11.** Provide 8-bit PWM output with the analogWrite() function.
- **SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK).** These pins support SPI communication using the SPI library.

- **LED: 13.** There is a built-in LED connected to digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.

The Uno has 6 analog inputs, labeled A0 through A5, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though is it possible to change the upper end of their range using the AREF pin and the analogReference() function. Additionally, some pins have specialized functionality:

- **TWI: A4 or SDA pin and A5 or SCL pin.** Support TWI communication using the Wire library.

There are a couple of other pins on the board:

- **AREF.** Reference voltage for the analog inputs. Used with analogReference().
- **Reset.** Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board.

See also the mapping between Arduino pins and ATmega328 ports. The mapping for the Atmega8, 168, and 328 is identical.

11. Communication

The Arduino Uno has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers. The ATmega328 provides UART TTL (5V) serial communication, which is available on digital pins 0 (RX) and 1 (TX). An ATmega16U2 on the board channels this serial communication over USB and appears as a virtual com port to software on the computer. The '16U2 firmware uses the standard USB COM drivers, and no external driver is needed. However, on Windows, a .inf file is required. The Arduino software includes a serial monitor which allows simple textual data to be sent to and from the Arduino board. The RX and TX LEDs on the board will flash when data is being transmitted via the USB-to-serial chip and USB connection to the computer (but not for serial communication on pins 0 and 1).

A SoftwareSerial library allows for serial communication on any of the Uno's digital pins.

The ATmega328 also supports I2C (TWI) and SPI communication. The Arduino software includes a Wire library to simplify use of the I2C bus; see the documentation for details. For SPI communication, use the SPI library.

12. Programming

The Arduino Uno can be programmed with the Arduino software (download). Select "Arduino Uno from the **Tools > Board** menu (according to the microcontroller on your board). For details, see the reference and tutorials.

The ATmega328 on the Arduino Uno comes preburned with a bootloader that allows you to upload new code to it without the use of an external hardware programmer. It communicates using the original STK500 protocol (reference, C header files).

You can also bypass the bootloader and program the microcontroller through the ICSP (In-Circuit Serial Programming) header; see these instructions for details.

The ATmega16U2 (or 8U2 in the rev1 and rev2 boards) firmware source code is available . The ATmega16U2/8U2 is loaded with a DFU bootloader, which can be activated by:

- On Rev1 boards: connecting the solder jumper on the back of the board (near the map of Italy) and then resetting the 8U2.
- On Rev2 or later boards: there is a resistor that pulling the 8U2/16U2 HWB line to ground, making it easier to put into DFU mode.

You can then use Atmel's FLIP software (Windows) or the DFU programmer (Mac OS X and Linux) to load a new firmware. Or you can use the ISP header with an external programmer (overwriting the DFU bootloader). See this user-contributed tutorial for more information.

13. Automatic (Software) Reset

Rather than requiring a physical press of the reset button before an upload, the Arduino Uno is designed in a way that allows it to be reset by software running on a connected computer. One of the hardware flow control lines (DTR) of the ATmega8U2/16U2 is connected to the reset line of the ATmega328 via a 100 nanofarad capacitor. When this line is asserted (taken low), the reset line drops long enough to reset the chip. The Arduino software uses this capability to allow you to upload code by simply pressing the upload button in the Arduino environment. This means that the bootloader can have a shorter timeout, as the lowering of DTR can be well-coordinated with the start of the upload. This setup has other implications. When the Uno is connected to either a computer running Mac OS X or Linux, it resets each time a connection is made to it from software (via USB). For the following halfsecond or so, the bootloader is running on the Uno. While it is programmed to ignore malformed data (i.e. anything besides an upload of new code), it will intercept the first few bytes of data sent to the board after a connection is opened. If a sketch running on the board receives one-time configuration or other data when it first starts, make sure that the software with which it communicates waits a second after opening the connection and before sending this data.

The Uno contains a trace that can be cut to disable the auto-reset. The pads on either side of the trace can be soldered together to re-enable it. It's labeled "RESET-EN". You may also be able to disable the auto-reset by connecting a 110 ohm resistor from 5V to the reset line; see this forum thread for details.

14. USB Overcurrent Protection

The Arduino Uno has a resettable polyfuse that protects your computer's USB ports from shorts and overcurrent. Although most computers provide their own internal

protection, the fuse provides an extra layer of protection. If more than 500 mA is applied to the USB port, the fuse will automatically break the connection until the short or overload is removed.

15. Physical Characteristics

The maximum length and width of the Uno PCB are 2.7 and 2.1 inches respectively, with the USB connector and power jack extending beyond the former dimension. Four screw holes allow the board to be attached to a surface or case. Note that the distance between digital pins 7 and 8 is 160 mil (0.16"), not an even multiple of the 100 mil spacing of the other pins.

Hook your Arduino up to GSM/GPRS cell phone network with GPRS shield! You can use your Arduino/Seeeduino or other main boards to dial a phone number or send a text to your friend via easy to use AT commands now. This new version features a quad-band low power consumption GSM/GPRS module SIM900 as well as a compact PCB antenna. Meanwhile, improvements on interfaces and basic circuit have been taken to make it more concise and reliable. And there're two choices for you to communicate GPRS shield with the main board -- UART or SoftwareSerial.

4.5. Version

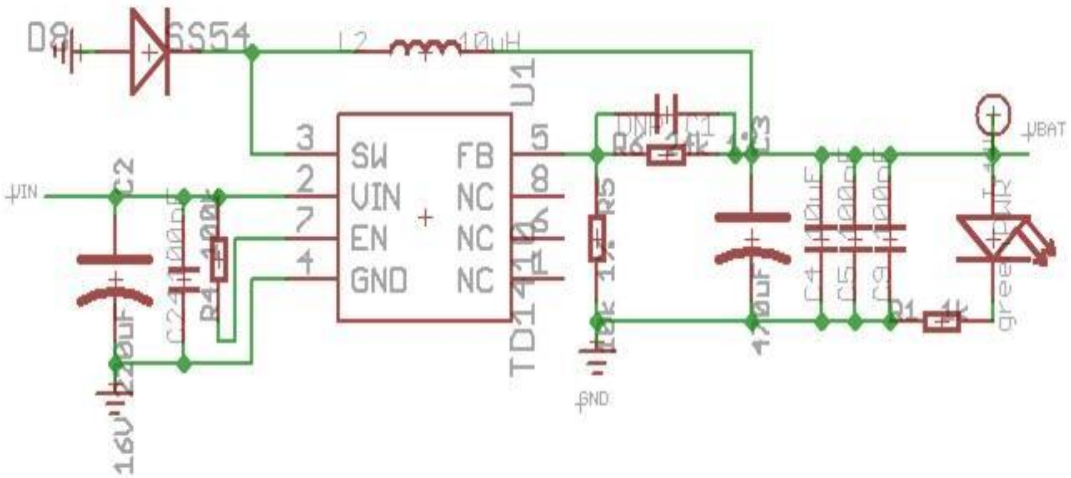
Revision	Descriptions	Release
v0.9b	Initial public release (beta)	Mar 3, 2011
v1.2	Added software port to power on/off of SIM90	Dec 2, 2011
v1.4	Re-design the power source circuit, re-lay the PCB layout	Aug 30, 2012
v2.0	Quad band support and re-design PCB antenna	Feb 3, 2013
v3.0	Change arduino socket to the latest Arduino Uno standard	Mar 20, 2015

What's the difference between V2.0 and previous version?

- Appearance Change
    - V2.0 adopts a standard shield outline as well as a protective shell;
    - Duck antenna is replaced by a compact PCB antenna;
    - Mic and earphone interfaces are replaced by 2-in-1 headset jack on V2.0.
- Power Circuitry Change

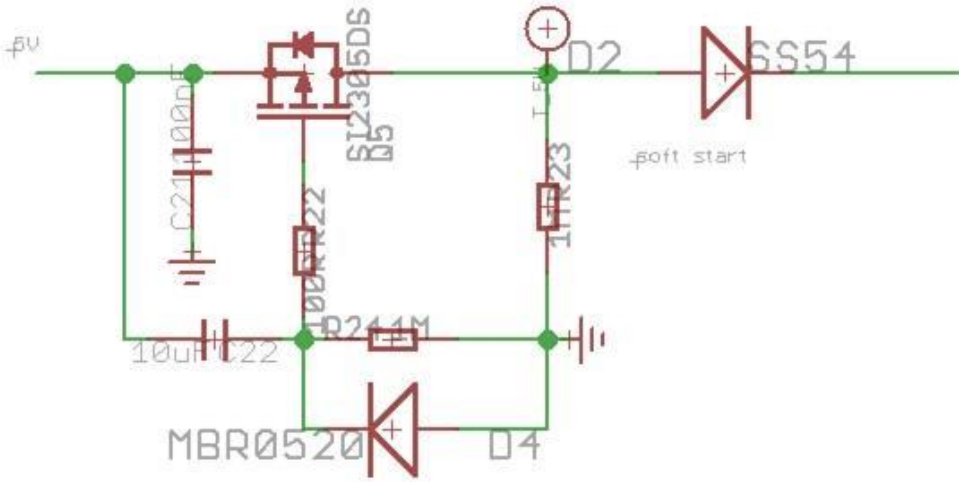
Replace the original LDO circuitry with DC-DC circuitry. Heat dissipation gets lower and efficiency gets higher up to 80%+. Meanwhile, the EXT_PWR jack on V1.0 was

removed. V2.0 can draw current directly from Arduino now without additional 5V



adapter.

- Soft Start Circuitry
    - Soft start circuitry is added in the new version to smooth out the power shock at the moment the shield turns on, preventing the shield from unexpected reset issue.



- Antenna Revision
    - The maximum transit power of SIM900 is 30dBm(1w). However the output power of V1.0 is only 0.4W. In this new version, transit power is turned up to 29dBm above(0.8w+), giving you more reliable and firm signal transmission.

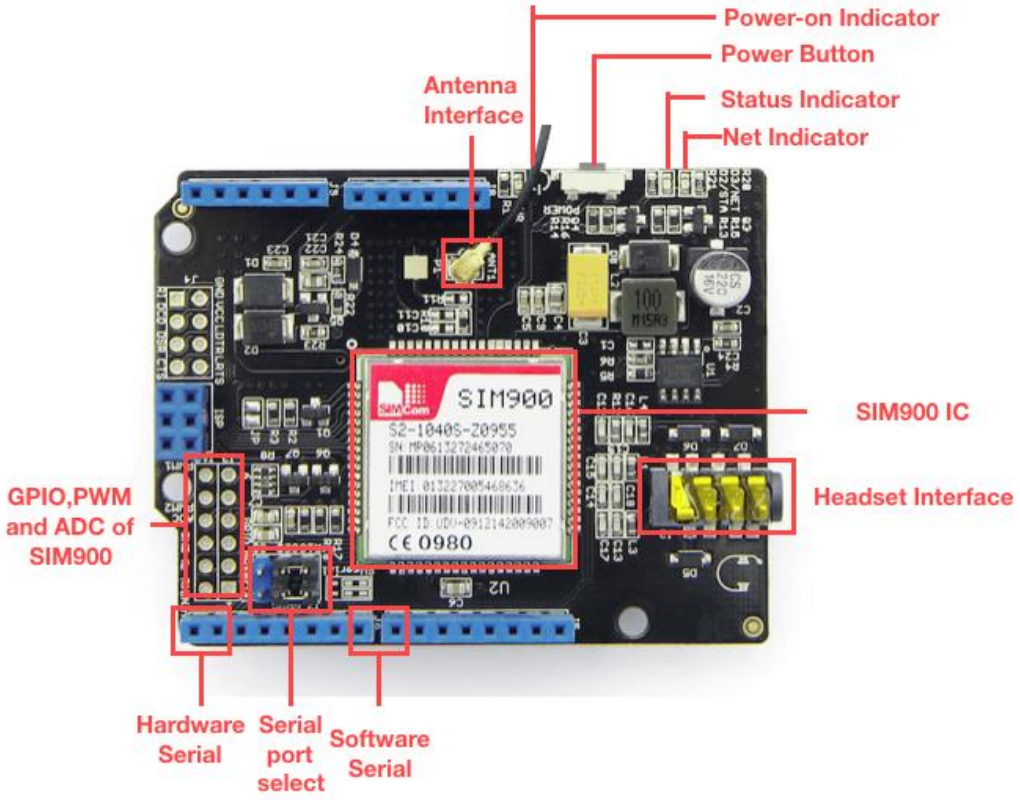
Specifications

Item	Value
Compatible	Arduino UNO/Seeeduino directly ; Other main board via jumpers
Selectable interface	UART, Software serial
Quad band support	850/900/1800/1900MHz
Communication support	Standard - GSM 07.07 & 07.05 and Enhanced - SIMCOM AT Commands
Operation temperature	-40°C to +85 °C
Protocol support	0710 MUX protocol, embedded TCP/UDP protocol, FTP/HTTP, FOTA, MMS, embedded AT
Certification of SIM900	CE, IC, FCC, ROHS, PTCRB, GCF, ICASA, REACH, AT&T
Dimensions	68.58 * 53.34mm
Power supply	5v via 5V pin, 6.5~12v via Vin pin

Cautions

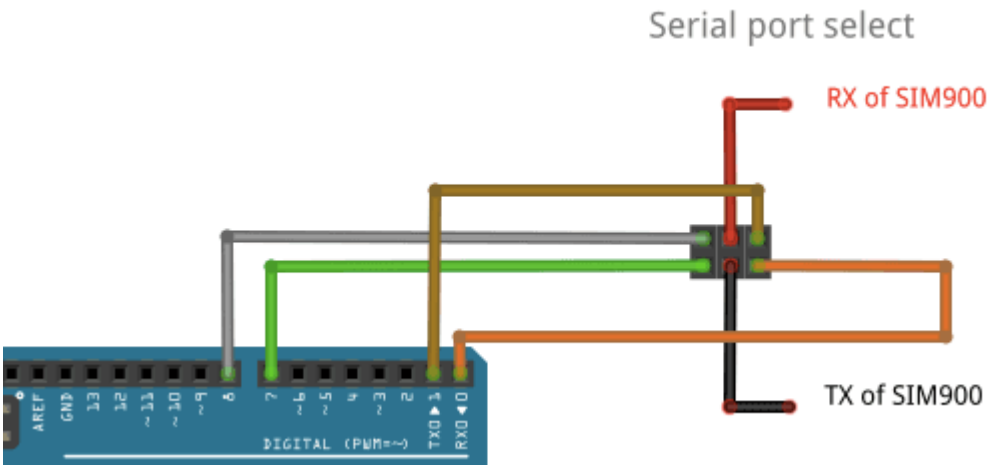
- Make sure your SIM card is activated.
- GPRS Shield doesn't come with ESD precautions. Take special care when handling it in dry weather.

Hardware Overview

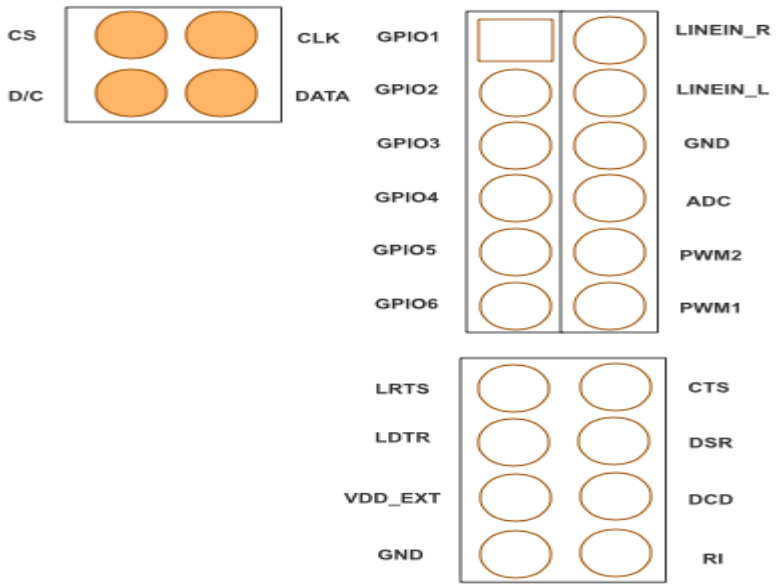


- The factory default setting for the GPRS Shield UART is 19200 bps 8-N-1. (Can be changed using AT commands).
- Serial port select
    - There're two choices for you to communicate GPRS shield with the main board while plugging the two jumpers on to the SWserial or HWserial positions. If using SWserial, D7 and D8 will be used by

SIM900 of GPRS Shield; if using HWserial, D0 (RX) and D1 (TX) will be used.



- Power on/off by D9
    - Unsolder pad JP default. Solder it if you wanna use software to power on/off the GPRS shield. Set the D9 to a high level, it means the button is pressing. The pad of JP besides the ISP port.
- Breakout of SIM900
    - Some pins of SIM900 are break out beside the ISP port, there're LINEIN_R, LINEIN_L, ADC, PWM1, PWM2, GPIO1~GPIO6, GND, FW_update (DISP_CLK, DISP_DATA, DISP_D/C, DISP_CS), RI, DCD, DSR, CTS, VDD_EXT, LDTR, LRTS. And those pins are pointed from SIM900 without any setting.



- RTC battery holder
    - It can provide 3v volts to VRTC of SIM900 from CR1220 battery.
- Power
    - Replace the original LDO circuitry with DC-DC circuitry -- TD1410. Heat dissipation gets lower and efficiency gets higher up to 80%+. Meanwhile, the output can up to 4.15V/2A. And there're two input of power supply: 5v pin: Soft start circuitry is added in the new version to smooth out the power shock at the moment the shield turns on, preventing the shield from unexpected reset issue. More detailed changes please refer to Related Reading : Version
    - Vin pin: The range of input voltage are between 6.5v to 12v.
- Antenna
    - The type of Antenna connector is IPEX, and the maximum transit power of SIM900 is 30dBm(1w). More information please see the Specification of GPRS Antenna.
- LED Status Description

LED	**Status**	**Function**
Power-on indicator(Green)	Off	Power of GPRS Shield is off
	On	Power of GPRS Shield is on
Status Indicator(Red)	Off	Power off
	On	Power on
Net indicator(Green)	Off	SIM900 is not working
	64ms On/800ms Off	SIM900 does not find the network
	64ms On/3000ms Off	SIM900 finds the network

	64ms On/300ms Off	**GPRS communication**

Getting Started

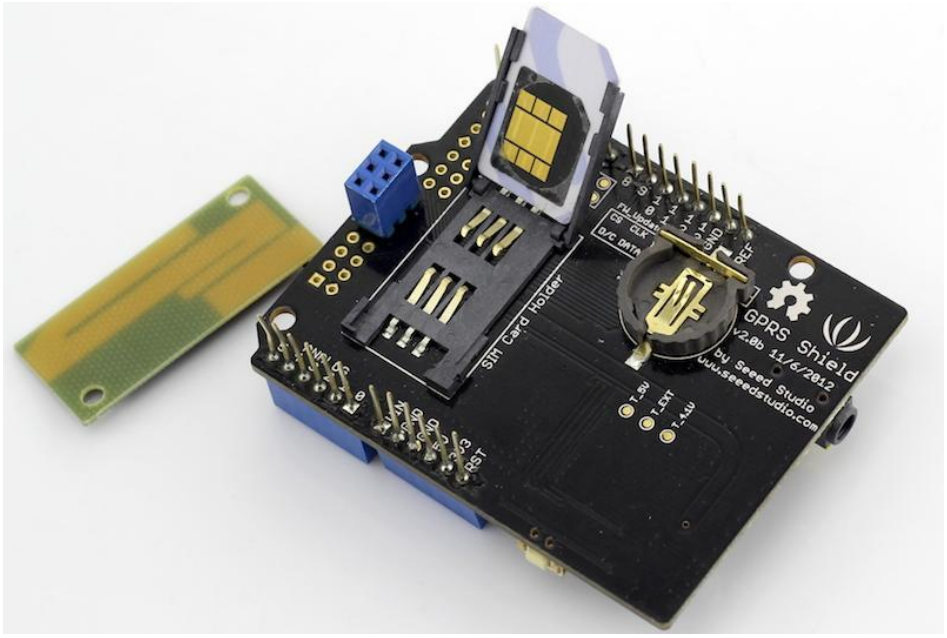
Getting Fun With AT Commands

As you receive the GPRS Shield, what would be the first thing you want to do with it? Send out a text (SMS)? Or call up someone (headset required)? You can do all of this by talking to the GPRS Shield using AT Commands - a special language that it understands.

AT Commands are simple textual commands sent to the GPRS modem over its serial interface (UART), so you can use any serial terminal software to communicate with it.

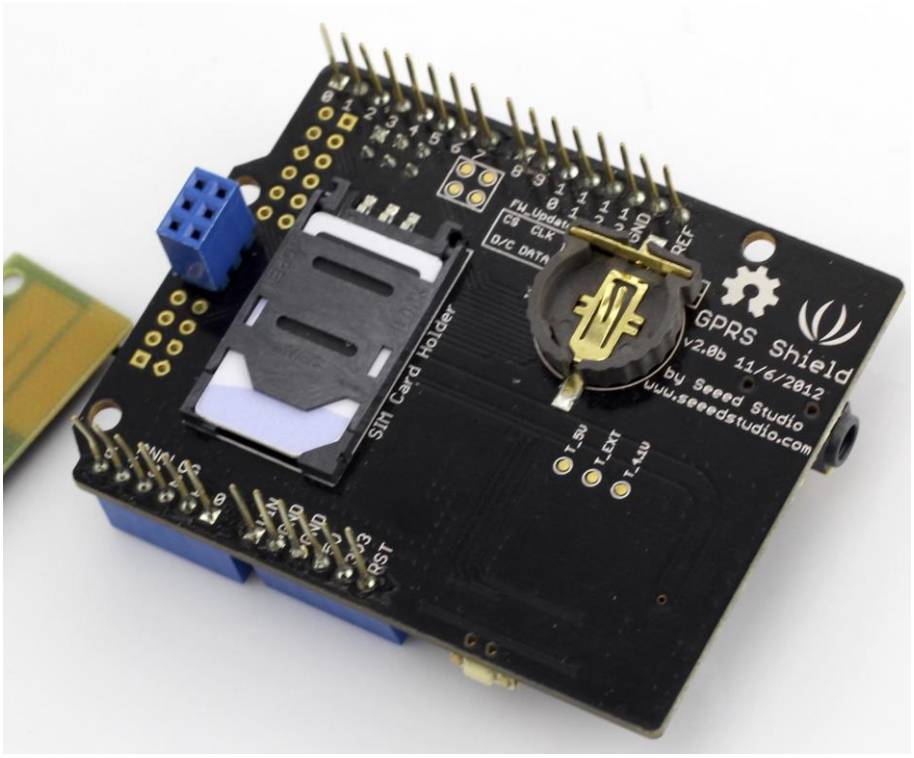
Hardware installation

- **Insert an activated SIM card into SIM Card Holder** - 6 Pin Holder for SIM Cards. Both 1.8 volts and 3.0 volts SIM Cards are supported by SIM900 - voltage type of the SIM card is automatically detected.
    - Insert the SIM Card into the holder

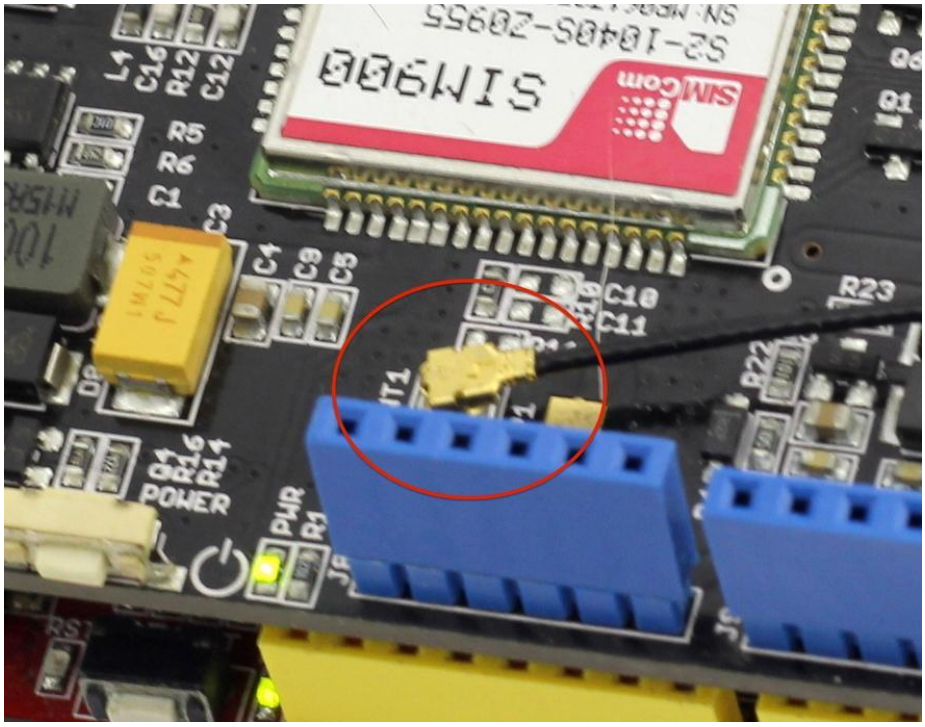


    - Lock the SIM Card Holder

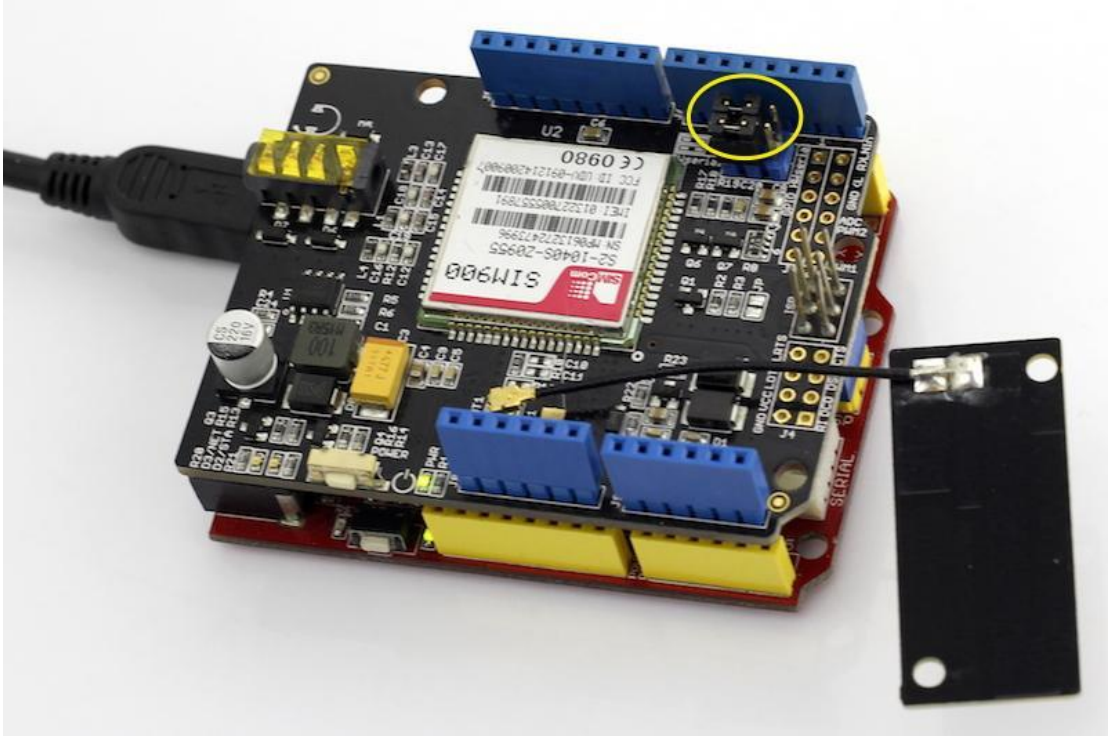
- **Make sure the antenna pad buckled properly**



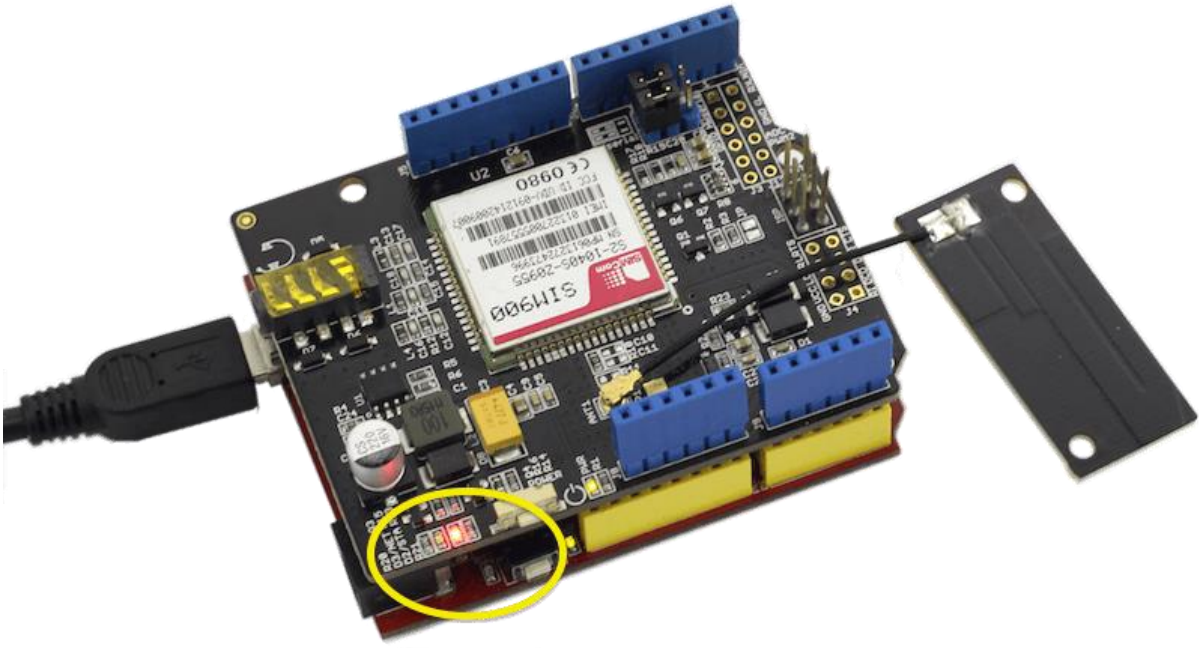
- **Communication port configuration** The GPRS shield can be controlled via hardware serial port or software serial port of Arduino. Here we use the software serial port as default. Choose it by inserting jumper caps as



below.



- **Plug into Arduino** - The GPRS Shield, like any other well designed shield, is stackable.
- **Power up Arduino** - Power up Arduino by USB cable or DC Jack. The Power-on indicator LED should light up once connected.



Software

Let's have a fun to control the GPRS shield with AT commands.

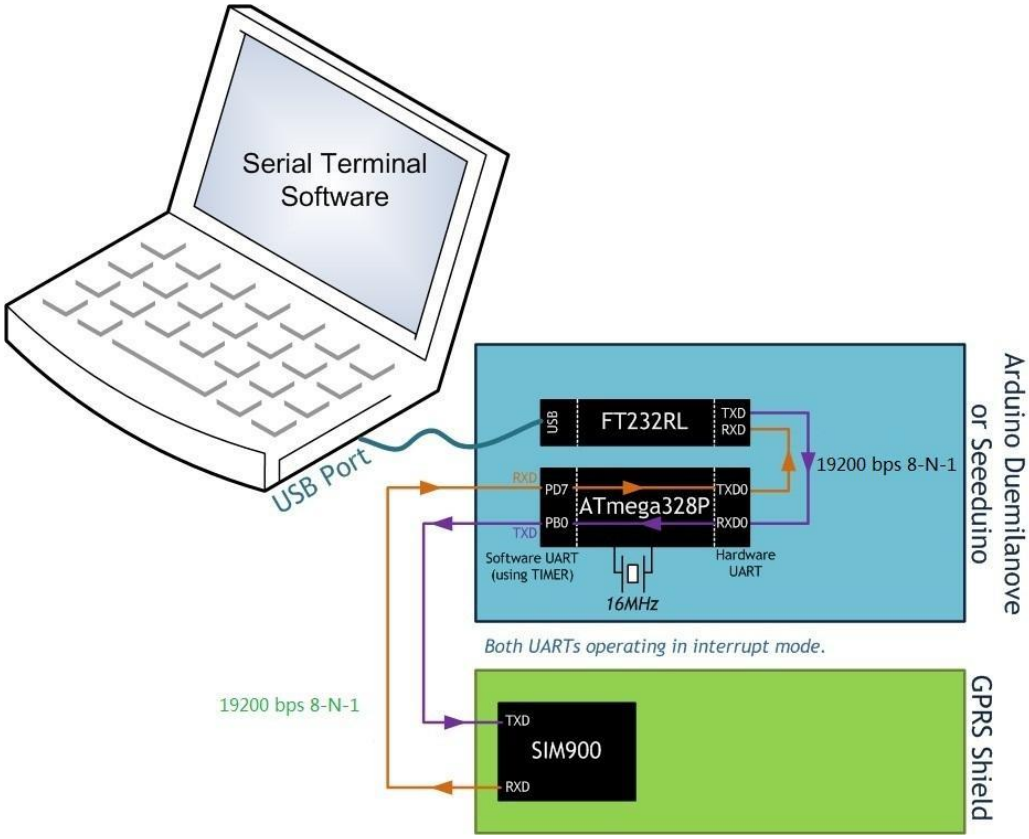
The GPRS Shield comes with all accessories that you need to get started with sending data over the GSM network except an Arduino board and a GSM SIM Card. If you want to make voice calls, you would also require a headset with microphone.

Step 1: Creating a test setup for the GPRS Shield

!!!note: Almost all the AT commands should be sent by following with **carriage return** and you need to select the "+CR"option in the serial port terminal.

To do experiment with AT commands, you would require a way to power up and communicate with your GPRS Shield. The best way to do this using an Arduino Duemilanove board described below. The same steps are applicable for Seeeduino or Seeeduino Stalker.

- Follow the previous hardware installation steps to set up the hardware system;
- Make sure the GPRS_TX & GPRS_RX jumpers on the GPRS Shield are mounted in SWSerial position - so GPRS_TX will be connected to D7(RX) and GPRS_RX to D8(TX).
- Connect the Arduino Duemilanove (or Seeeduino) to your computer by a USB cable.
- The ATmega328P microcontroller on Duemilanove board has only one UART which is used for communicating with the PC. What we need is an Arduino Sketch running inside the ATmega328P that would emulate a second serial port (UART) by using software serial on the digital pins D8 and D7. All the communication will go through the software serial port and the actual hardware serial port. By doing this, all the data coming from the computer (connected to the actual hardware UART) would be routed to the GPRS Shield (connected to software UART). Then we would be able to issue AT commands to control the GPRS Shield. The block diagram outlining this scheme is shown below.



For developing such a program, we need to use the SoftwareSerial library. Here is the demo code.

```
//Serial Relay - Arduino will patch a
//serial link between the computer and the GPRS Shield
//at 19200 bps 8-N-1
//Computer is connected to Hardware UART
//GPRS Shield is connected to the Software UART

#include <SoftwareSerial.h>

SoftwareSerial GPRS(7, 8);
unsigned char buffer[64]; // buffer array for data recieve over serial port
int count=0;     // counter for buffer array
void setup()
{
    GPRS.begin(19200);               // the GPRS baud rate
    Serial.begin(19200);             // the Serial port of Arduino baud rate.

}

void loop()
{
    if (GPRS.available())              // if date is comming from
softwareserial port ==> data is comming from gprs shield
    {
        while(GPRS.available())          // reading data into char array
        {
            buffer[count++]=GPRS.read();     // writing data into array
            if(count == 64)break;
        }
```

```
        Serial.write(buffer,count);            // if no data transmission
ends, write buffer to hardware serial port
        clearBufferArray();              // call clearBufferArray function to
clear the storaged data from the array
        count = 0;                       // set counter of while loop to zero


    }
    if (Serial.available())            // if data is available on
hardwareserial port ==> data is comming from PC or notebook
    GPRS.write(Serial.read());       // write it to the GPRS shield
}
void clearBufferArray()              // function to clear buffer array
{
    for (int i=0; i<count;i++)
    { buffer[i]=NULL;}                  // clear all index of array with
command NULL
}
```

- Upload the sketch to the Arduino board. If you do not know how to upload the code, please follow the instruction.
- Download and fire up serial tool if you don't have one. Choose the correct COM port for Arduino, and set it to operate at **19200** 8-N-1 and then click "Open COM". You can also use AT Command Tester to send AT commands. Please click here if you are interesting in it.
- Power up the SIM900 by pressing the power button in about 2 seconds. The red LED will be on. The green one beside it will blink. If the shield join the network sucessfully, the green LED will blink every 3 seconds.
- You should find the message on the serial monitor as below which is sent by the SIM900 to inform you it has joined the network.

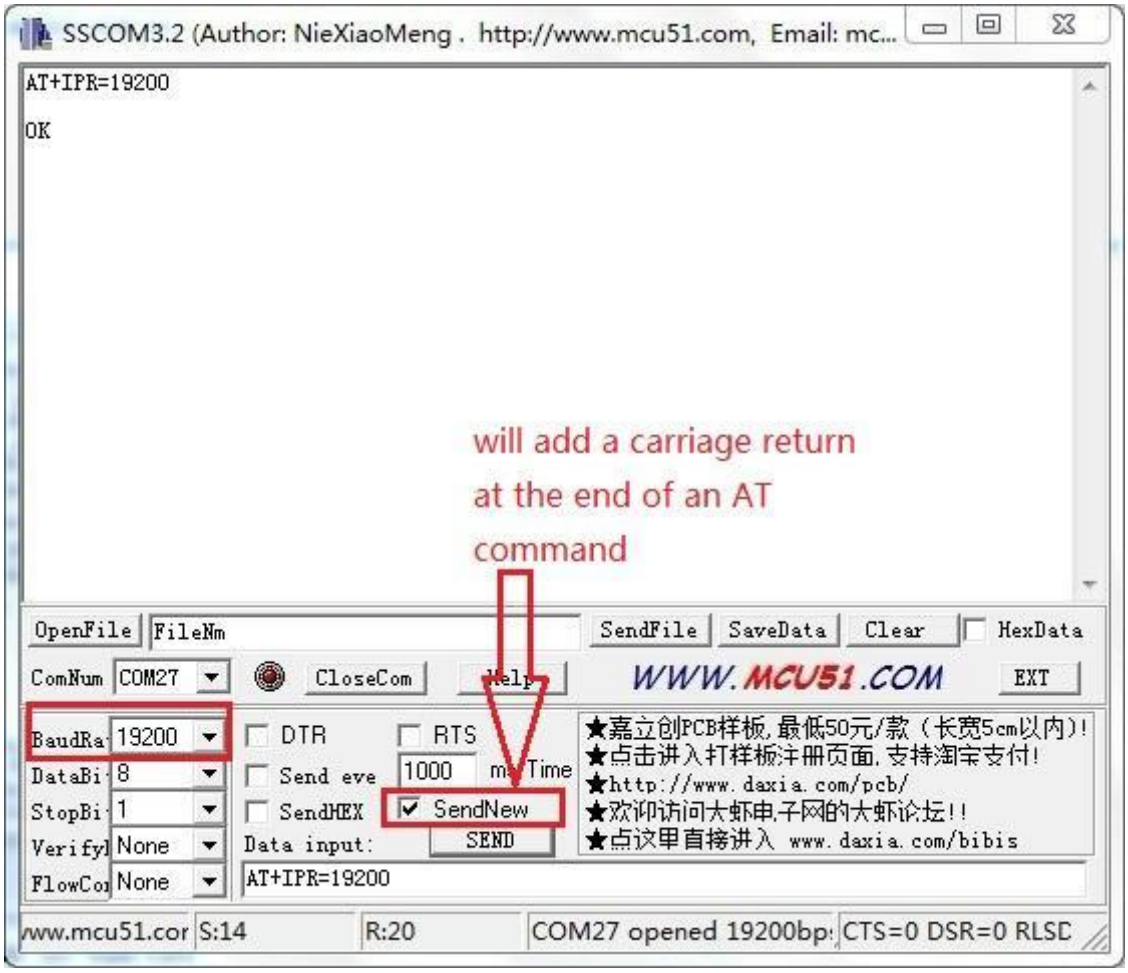
```
RDY

+CFUN: 1

+CPIN: READY

Call Ready
```

If you can not see the messages in the serial monitor, you should click the "send new" option that will add carriage return at the end of AT command and then send AT command "AT+IPR=19200" to set the baud rate of the SIM900.

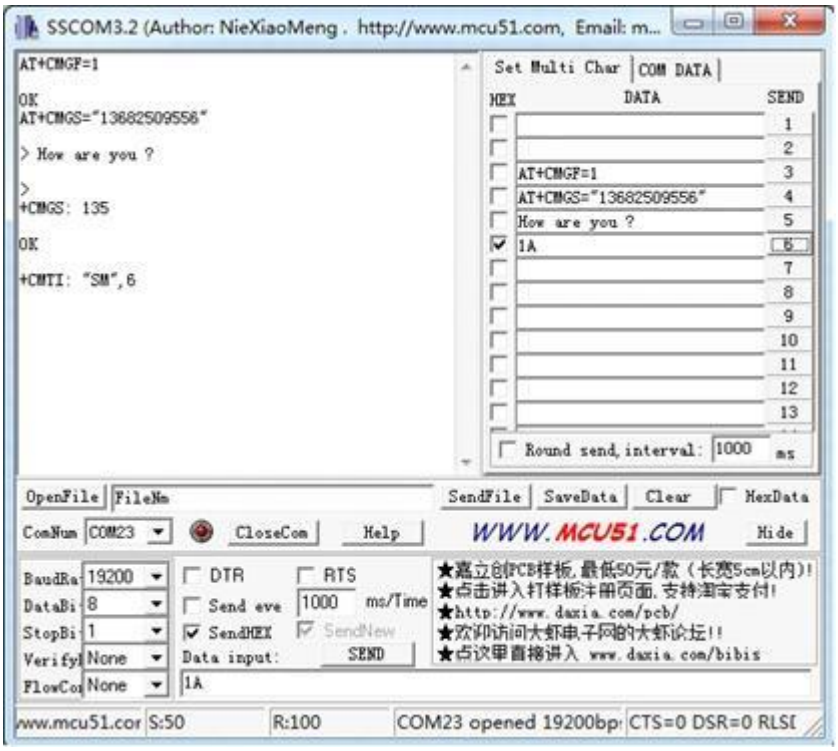


Step 2: Sending a text message (SMS)

Now that our test setup is ready, let's play around with some AT Commands manually before moving on to program the Arduino to do this. Let's try sending an SMS to start.

- Create the setup as described in Step 1 above.
- Through your serial terminal software, send **AT+CMGF=1** and press the Enter key. The GPRS Shield can send SMSes in two modes: Text mode and PDU (or binary) mode. Since we want to send out a human readable message, we will select the text mode. The GPRS Shield will respond with an **OK**.
- Click "send new" option and send **AT+CMGS="+18888888888"**. This will instruct the GPRS Shield to start to accept text for a new message, numbers mean the specified phone number(replace the number with the phone number of the target phone). The GPRS Shield will send a '**>**' to remind you to type the message. Please note that phone numbers in any AT Command must follow E.123 format format.

- Start to type your message, after finishing the typing, click "send hex" option and then send a hex: **1A**. The modem will accept the message and respond with an **OK**. A few moments later, the message should be received on the handset whose number you had specified. I sent "How are you ?". You can check the history by clicking "EXT". The commands history is listed below "Set Multi Char".



!!!note: If, in spite of following the steps as specified above, you're unable to receive the message on the target handset, you might need to set the SMS Message Center number. Send the following command between the AT+CMGF and AT+CMGS commands:AT+CSCA="+18888888888". Replace the phone number specified with the SMS Center number of your GSM Service Provider. The message center number is specific to each service provider. You can get the message center number by calling up the customer care center of the GSM Service Provider and asking them for it.

Step 3: Make a telephone call with the AT commands

- Restart the SIM900 if switching from sending texts to making phone calls.
- Replace the phone number with your target number in the command “ATD18888888888;”(without the quotes) and press Enter key to send it out. If it succeds, a message "ATH OK" will show up as the picture below. Otherwise, "No CRRLIER" will show up instead. The reason might be unexistent phone number or incorrect command format.

Step 4: Exploring further

Now that you have gotten a taste of how the AT Commands work, you can try out other commands before moving on to develop sketches for Arduino by using the GPRS Shield.

This involves to create a sketch for sending out these same sequence of AT Commands (on your behalf) out the serial port to the GPRS Shield to perform the same task of sending and SMS, making a call or sending data over a GPRS connection.

You can go through the AT Commands reference manual to figure out the sequence of commands. If you develop an Arduino sketch, you find that the GPRS Shield doesn't act as what you expect, then you will need to check your AT Commands and their sequence. To do this, reload the serial relay sketch attached above in the getting started section into ATmega328P and type out the AT Commands manually and check the output. The responses sent back by the GPRS Shield will help you debug the AT Command sequence.

!!!note: A C program to perform the same task has also been developed and attached: Softuart relay atmega328p.zip.

The program was developed on a Windows PC.AVRStudio4 was used as the IDE and WinAVR was used as the compiler. The ZIP file contains an AVRStudio4 Project. The C compiler (WinAVR) will generate an Intel Hex (.hex). To upload this .hex file into an Arduino board outside of Arduino IDE would require a program which is able to communicate with the Arduino boards bootloader. XLoader is such a program which runs on Windows can upload .hex files generated by various compiler into an Arduino Board.

SoftwareSerial library Notes

With Arduino you should be able to use the SoftwareSerial library included with the distribution (instead of NewSoftSerial). However, you must be aware that the buffer reserved for incoming messages are hardcoded to 64 bytes in the library header, "SoftwareSerial.h":

```
define _SS_MAX_RX_BUFF 64 // RX buffer size
```

This means that if the GPRS module receive more data than the buffer, you are likely to lose it with a buffer overflow! For instance, reading out an SMS from the module with "AT+CMGR=xx" (xx is the message index), you might not even see the message part because the preceding header information (like telephone number and time) takes up a lot of space. The fix seems to be to manually change **_SS_MAX_RX_BUFF** to a higher value (but reasonable so you don't use all your memory!)

The Softwareserial library has the following limitations (taken from arduino page). If use multiple software serial ports, only one can receive data at a time. This means that if you try to add another serial device such as grove serial LCD you may get communication errors unless you craft your code carefully.

How To Send a Text Message or Dial a Phone Number Using AT Commands

Tn this example we will create an Arduino sketch to allow you to either send a text message or dial a phone number for a voice call when you either type 't' or 'd' respectively in the Arduino's serial comm window.

AT Commands For Sending a Text Message

Looking at the SIM900 Commands Set we can see that to send a text message we first have to set the SMS format, this is one using the AT+CMGF command.

The AT+CMGF command can be one of two values: 0, or 1 for "PDU mode" and "text mode" respectively. In "text mode", SMS messages are represented as readable text. In "PDU mode", SMS messages are represented as binary strings encoded in hexadecimal characters. We will use "text mode" in this example, so the command we need to the send to the GPRS shield is:

AT+CMGF=1\r

Now that we have set the SMS format we can proceed to send the text message. To send the text message we use the AT+CMGS command. According to the SIM900 AT Command Set, the CMGS command follows this format:

AT+CMGS=<da>[,<toda>]<CR>

where <da> is the destination address (phone number), <toda> is the optional type of destination address, and <CR> is the carriage return '\r' character. Once we have send this command the GPRS shield will reply with the '>' character signaling us to enter the actual message.

!!!Note The phone number must include the country code, e.g. for a U.S. phone number (555)123-4567 the <da> value must be +15551234567

Command(s) For a Voice Call

As per the SIM900 AT Command Set sheet, the command used to dial a number for a voice call is "ATD+xxxxxxxxxx;" where "xxxxxxxxxx" is the phone number with country code included. A semicolon must be send to set up a voice call, omitting it will set up a data or fax call.

The Arduino code to send a simple text message or dial a voice call is shown below. Don't forget to change the phone number "xxxxxxxxxx" and message "How are you today?" to your own values.

Arduino Code:

```
  #include <SoftwareSerial.h>


  SoftwareSerial gprsSerial(7,8);

  void setup()
  {
    gprsSerial.begin(19200); // GPRS shield baud rate
    Serial.begin(19200);
    delay(500);
  }

  void loop()
  {

    if (Serial.available()) // if there is incoming serial data
     switch(Serial.read()) // read the character
     {
       case 't': // if the character is 't'
         SendTextMessage(); // send the text message
         break;
       case 'd': // if the character is 'd'
         DialVoiceCall(); // dial a number
         break;

     }

    if (gprsSerial.available()){ // if the shield has something to say
      Serial.write(gprsSerial.read()); // display the output of the shield
    }
  }

  /*
  * Name: SendTextMessage
  * Description: Send a text message to a number
  */
  void SendTextMessage()
  {
    Serial.println("Sending Text...");
    gprsSerial.print("AT+CMGF=1\r"); // Set the shield to SMS mode
    delay(100);
    // send sms message, the phone number needs to include the country code
e.g. if a U.S. phone number such as (540) 898-5543 then the string must be:
    // +15408985543
    gprsSerial.println("AT+CMGS = \"+xxxxxxxxxx\"");
    delay(100);
    gprsSerial.println("How are you today?"); //the content of the message
    delay(100);
    gprsSerial.print((char)26);//the ASCII code of the ctrl+z is 26 (required
according to the datasheet)
    delay(100);
```

```
    gprsSerial.println();
    Serial.println("Text Sent.");
  }

  /*
  * Name: DialVoiceCall()
  * Description: Can call/dial a phone number
  */
  void DialVoiceCall()
  {
    gprsSerial.println("ATD+xxxxxxxxxx;");//dial the number, must include
country code
    delay(100);
    gprsSerial.println();
  }
```

4.6. FAQ

Here is the GPRS Shield FAQ that list the frequently asked questions, please read it before use:

- Why does GPRS Shield can not work with Stalker v2.1/2.0 via software serial port(D7/D8) ?

  Talker v2.1/2.0 has used the Digital Pin(D7/D8) , you need to connect GPRS_TX/GPRS_RX to other Digital Pin as software serial port. And this problem had been solved at Stalker 2.2 version.

- Why does the I2C can't be used when I assembled the GPRS Shield to Stalker or other Arduino boards ?

  Because GPRS Shield conflicts with other module via I2C . GPRS Shield doesn't use I2C port, you can disconnect the connection from SIM900 I2C port to A4/A5 .